ifm electronic

CE

ecomat mobile

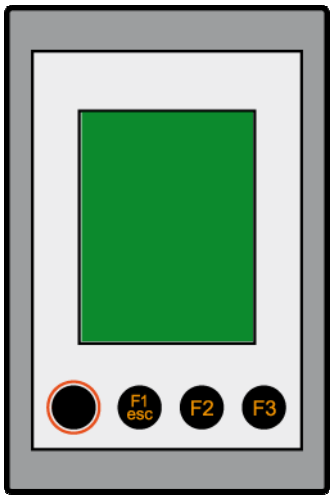
Systemhandbuch
PDM360compact Monitor

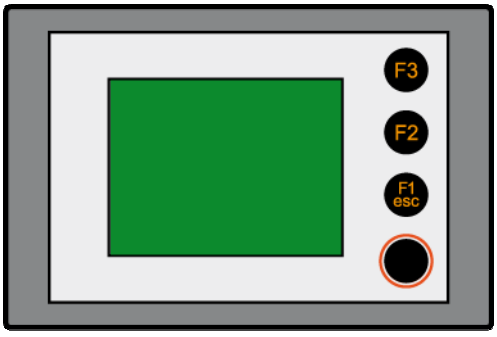
ecomat100

CR1052
CR1053
CR1055
CR1056

CoDeSys® V2.3
Target V04

Deutsch





7390676_00_DE 2012-07-12

© ifm electronic gmbh

Inhaltsverzeichnis

1 Über diese Anleitung 5
1.1 Was bedeuten die Symbole und Formatierungen? 5
1.2 Wie ist diese Anleitung aufgebaut? 6

2 Sicherheitshinweise 7
2.1 Wichtig! 7
2.2 Welche Vorkenntnisse sind notwendig? 8

3 Systembeschreibung 9
3.1 Angaben zum Gerät 9
3.2 Angaben zur Software (Linux | CoDeSys v2.3) 9
3.3 Steuerungskonfiguration 10

4 Konfigurationen 11
4.1 Geräteparameter einstellen (Setup) 11
4.1.1 Setup starten 12
4.1.2 Ethernet-Schnittstelle: IP-Adresse einstellen 13
4.1.3 Ethernet-Schnittstelle: Subnetzmaske einstellen 15
4.1.4 Startbild des PDM ändern 17
4.1.5 Echtzeituhr (RTC) einstellen 20
4.1.6 Funktion von Tasten, Drehgeber und LEDs prüfen 21
4.1.7 CAN-Schnittstelle prüfen 22
4.1.8 CoDeSys-Kommunikationsparameter einstellen 24
4.1.9 PDM-Setup verlassen, Gerät neu starten 27
4.2 Programmierschnittstellen 28
4.2.1 Programmierung über die Ethernet-Schnittstelle 29
4.2.2 Programmierung über die serielle Schnittstelle RS232 35
4.3 Zeichensätze (Fonts) installieren 36
4.3.1 Symbol-Dateien 38
4.4 Programmiersystem einrichten 39
4.4.1 Programmiersystem manuell einrichten 39
4.4.2 Programmiersystem über Templates einrichten 44
4.4.3 ifm-Demo-Programme 53
4.5 Erste Schritte 57
4.5.1 Fehlende Bibliotheken einfügen 57
4.5.2 Visualisierung erstellen 59
4.5.3 PLC-Programm erstellen 61
4.5.4 Projekt im Gerät speichern 62
4.6 Geräte-Update auf neue Software-Version 63
4.6.1 Geräte-Update vorbereiten 63
4.6.2 Geräte-Update durchführen 66

5 Begrenzungen und Programmierhinweise 69
5.1 Leistungsgrenzen des Geräts 69
5.1.1 CPU-Frequenzen 69
5.1.2 Verhalten des Watchdog 70
5.1.3 Begrenzungen beim PDM360compact 71
5.1.4 Visualisierungsgrenzen 72

© ifm electronic gmbh

5.2 Programmierhinweise für CoDeSys-Projekte 75
5.2.1 FB, FUN, PRG in CoDeSys 76
5.2.2 Bibliotheken 77
5.2.3 Arbeitsreihenfolge 78
5.2.4 Speichern 78
5.2.5 ifm-Maintenance-Tool nutzen 79

6 Variable 80

6.1 Retain-Variable 81
6.1.1 Sichern von Retain-Variablen 81
6.1.2 Rücklesen von Retain-Variablen 82
6.2 Netzwerkvariablen 82

7 CAN einsetzen 83

7.1 Allgemeines zu CAN 83
7.1.1 Topologie 84
7.1.2 CAN-Schnittstellen 85
7.1.3 Verfügbare CAN-Schnittstellen und CAN-Protokolle 85
7.1.4 System-Konfiguration 87
7.1.5 Software für CAN und CANopen 87
7.2 Physikalische Anbindung des CAN 88
7.2.1 Netzaufbau 88
7.2.2 CAN-Buspegel 89
7.2.3 Busleitungslänge 90
7.2.4 Leitungsquerschnitte 91
7.3 CAN-Datenaustausch 92
7.3.1 Hinweise 93
7.3.2 Daten empfangen 95
7.3.3 Daten senden 95
7.4 Beschreibung der CAN-Standardbausteine 96
7.4.1 CAN_x_RECEIVE 97
7.4.2 CAN_x_TRANSMIT 99
7.5 ifm-CANopen-Bibliotheken 101
7.5.1 Technisches zu CANopen 101
7.5.2 Bibliotheken für CANopen 141
7.6 CAN-Fehler und Fehlerbehandlung 170
7.6.1 CAN-Fehler 171
7.6.2 Aufbau einer EMCY-Nachricht 173
7.6.3 Übersicht CANopen Error-Codes 175

8 Ein-/Ausgangs-Funktionen 178

8.1 Eingangswerte verarbeiten 178
8.1.1 TOGGLE 179
8.2 Analoge Werte anpassen 180
8.2.1 NORM_DINT 181
8.2.2 NORM_REAL 183

9 Daten verwalten 185

9.1 Systemzeit lesen / schreiben 185
9.1.1 PDM_READ_RTC 186
9.1.2 PDM_SET_RTC 188
9.2 Datenzugriff und Datenprüfung 190
9.2.1 DEVICE_KERNEL_VERSION2 191
9.2.2 DEVICE_RUNTIME_VERSION 192
9.2.3 DEVICE_TS 193

© ifm electronic gmbh

9.3 Dateien verwalten ... 194
9.3.1 Dateifunktionen ... 194
9.3.2 Asynchrone Dateifunktionen ... 233

10 SPS-Zyklus optimieren 241
10.1 Zykluszeit steuern ... 241
10.1.1 GET_PENALTY ... 242
10.1.2 OPT_SCHEDULER ... 244

11 LED, Buzzer, Visualisierung 249
11.1 LED ansteuern ... 249
11.1.1 PDM_LED_OFF ... 250
11.2 Visualisierung verwalten ... 251
11.2.1 PDM_MAIN ... 252
11.2.2 PDM_MAIN_MAPPER ... 255
11.2.3 PDM_PAGECONTROL ... 259
11.2.4 CURRENTVISU_TO_NO ... 261
11.2.5 PAGENO_TO_CURRENTVISU ... 263
11.2.6 Bibliothek Instrumente ... 264
11.3 Eingabefelder in Visualisierung bearbeiten ... 271
11.3.1 INPUT_INT ... 272
11.3.2 INPUT_DINT ... 277
11.3.3 INPUT_REAL ... 280
11.3.4 INIT_VAR_RETAIN_INT ... 283
11.3.5 INIT_VAR_RETAIN_DINT ... 285
11.3.6 INIT_VAR_RETAIN_REAL ... 287

12 Anhang 289
12.1 Fehler und Störungen beheben ... 289
12.2 CANopen-Tabellen ... 290
12.2.1 IDs (Adressen) in CANopen ... 290
12.2.2 Aufbau von CANopen-Meldungen ... 291
12.2.3 Bootup-Nachricht ... 296
12.2.4 Netzwerk-Management (NMT) ... 297
12.2.5 CANopen Error-Code ... 301
12.3 Visualisierungen im Gerät ... 304
12.3.1 Grundsätzliches ... 304
12.3.2 Empfehlungen für Bedienoberflächen ... 305
12.3.3 Grundlegende Informationen zu Bitmap-Grafiken ... 319
12.4 Übersicht der verwendeten Dateien und Bibliotheken ... 322
12.4.1 Dateien und Bibliotheken im Gerät installieren ... 322
12.4.2 Allgemeine Übersicht ... 323
12.4.3 Wozu dienen die einzelnen Dateien und Bibliotheken? ... 325

13 Begriffe und Abkürzungen 333

14 Index 350

15 ifm weltweit • ifm worldwide • ifm à l'échelle internationale 357

© ifm electronic gmbh

1 Über diese Anleitung

Inhalt:

Was bedeuten die Symbole und Formatierungen? 5
Wie ist diese Anleitung aufgebaut? 6

202

Im ergänzenden "Programmierhandbuch CoDeSys V2.3" erhalten Sie weitergehende Informationen über die Nutzung des Programmiersystems "CoDeSys for Automation Alliance". Dieses Handbuch steht auf der **ifm**-Homepage als kostenloser Download zur Verfügung:
a) → www.ifm.com > Land wählen > [Service] > [Download] > [Steuerungssysteme]
b) → **ecomat*mobile***-DVD "Software, tools and documentation"

Niemand ist vollkommen. Wenn Sie uns Verbesserungsvorschläge zu dieser Anleitung melden, erhalten Sie von uns ein kleines Geschenk als Dankeschön.

© Alle Rechte bei **ifm electronic gmbh**. Vervielfältigung und Verwertung dieser Anleitung, auch auszugsweise, nur mit Zustimmung der **ifm electronic gmbh**.

Alle auf unseren Seiten verwendeten Produktnamen, -Bilder, Unternehmen oder sonstige Marken sind Eigentum der jeweiligen Rechteinhaber:
- AS-i ist Eigentum der AS-International Association, (→ www.as-interface.net)
- CAN ist Eigentum der CiA (CAN in Automation e.V.), Deutschland (→ www.can-cia.org)
- CoDeSys™ ist Eigentum der 3S – Smart Software Solutions GmbH, Deutschland (→ www.3s-software.com)
- DeviceNet™ ist Eigentum der ODVA™ (Open DeviceNet Vendor Association), USA (→ www.odva.org)
- IO-Link® (→ www.io-link.com) ist Eigentum der →PROFIBUS Nutzerorganisation e.V., Deutschland
- Microsoft® ist Eigentum der Microsoft Corporation, USA (→ www.microsoft.com)
- PROFIBUS® ist Eigentum der PROFIBUS Nutzerorganisation e.V., Deutschland (→ www.profibus.com)
- PROFINET® ist Eigentum der →PROFIBUS Nutzerorganisation e.V., Deutschland
- Windows® ist Eigentum der →Microsoft Corporation, USA

1.1 Was bedeuten die Symbole und Formatierungen?

2979
203

Folgende Symbole oder Piktogramme verdeutlichen Ihnen unsere Hinweise in unseren Anleitungen:

⚠ WARNUNG

Tod oder schwere irreversible Verletzungen sind möglich.

⚠ VORSICHT

Leichte reversible Verletzungen sind möglich.

ACHTUNG

Sachschaden ist zu erwarten oder möglich.

	Wichtige Hinweise auf Fehlfunktionen oder Störungen
	Weitere Hinweise
► ...	Handlungsaufforderung
> ...	Reaktion, Ergebnis
→ ...	"siehe"
abc	Querverweis

[...]	Bezeichnung von Tasten, Schaltflächen oder Anzeigen
	PDM-Drehgeber: Drehknopf drehen
	PDM-Drehgeber: Drehknopf drücken PDM-Kreuzwippe: zentrale Taste drücken

1.2 Wie ist diese Anleitung aufgebaut?

204

Diese Dokumentation ist eine Kombination aus verschiedenen Anleitungstypen. Sie ist eine Lernanleitung für den Einsteiger, aber gleichzeitig auch eine Nachschlageanleitung für den versierten Anwender.

Und so finden Sie sich zurecht:

- Um gezielt zu einem bestimmten Thema zu gelangen, benutzen Sie bitte das Inhaltsverzeichnis.
- Mit dem Stichwortregister "Index" gelangen Sie ebenfalls schnell zu einem gesuchten Begriff.
- Am Anfang eines Kapitels geben wir Ihnen eine kurze Übersicht über dessen Inhalt.
- Abkürzungen und Fachbegriffe → Anhang.

Bei Fehlfunktionen oder Unklarheiten setzen Sie sich bitte mit dem Hersteller in Verbindung:
→ www.ifm.com > Land wählen > [Kontakt].

Wir wollen immer besser werden! Jeder eigenständige Abschnitt enthält in der rechten oberen Ecke eine Identifikationsnummer. Wenn Sie uns über Unstimmigkeiten unterrichten wollen, dann nennen Sie uns bitte diese Nummer zusammen mit Titel und Sprache dieser Dokumentation. Vielen Dank für Ihre Unterstützung!

Im Übrigen behalten wir uns Änderungen vor, so dass sich Abweichungen vom Inhalt der vorliegenden Dokumentation ergeben können. Die aktuelle Version finden Sie auf der **ifm**-Homepage:
→ www.ifm.com > Land wählen > [Service] > [Download] > [Steuerungssysteme]
⇨ Unsere Online-Hilfen sind meist "tagesaktuell".
⇨ Die PDF-Handbücher aktualisieren wir nur in großen zeitlichen Abständen.

© ifm electronic gmbh

2 Sicherheitshinweise

Inhalt:

Wichtig! 7
Welche Vorkenntnisse sind notwendig? 8

213

2.1 Wichtig!

214

Mit den in dieser Anleitung gegebenen Informationen, Hinweisen und Beispielen werden keine Eigenschaften zugesichert. Die abgebildeten Zeichnungen, Darstellungen und Beispiele enthalten weder Systemverantwortung noch applikationsspezifische Besonderheiten.

Die Sicherheit der Maschine/Anlage muss auf jeden Fall eigenverantwortlich durch den Hersteller der Maschine/Anlage gewährleistet werden.

⚠ WARNUNG

Sach- oder Körperschäden sind möglich bei Nichtbeachten der Hinweise in dieser Anleitung!
Die **ifm electronic gmbh** übernimmt hierfür keine Haftung.

- ► Die handelnde Person muss vor allen Arbeiten an und mit diesem Gerät die Sicherheitshinweise und die betreffenden Kapitel dieser Anleitung gelesen und verstanden haben.
- ► Die handelnde Person muss zu Arbeiten an der Maschine/Anlage autorisiert sein.
- ► Beachten Sie die Technischen Daten der betroffenen Geräte!
  Das aktuelle Datenblatt finden Sie auf der **ifm**-Homepage:
  → www.ifm.com > Land wählen > [Datenblattsuche] > (Artikel-Nr.) > [Technische Daten im PDF-Format]
- ► Beachten Sie die Montage- und Anschlussbedingungen sowie die bestimmungsgemäße Verwendung der betroffenen Geräte!
  → mitgelieferte Montageanleitung oder auf der **ifm**-Homepage:
  → www.ifm.com > Land wählen > [Datenblattsuche] > (Artikel-Nr.) > [Betriebsanleitungen]

ACHTUNG

Der Treiberbaustein der seriellen Schnittstelle kann beschädigt werden!

Beim Trennen der seriellen Schnittstelle unter Spannung kann es zu undefinierten Zuständen kommen, die zu einer Schädigung des Treiberbausteins führen.

- ► Die serielle Schnittstelle nur im spannungslosen Zustand trennen!

Anlaufverhalten der Steuerung

Der Hersteller der Maschine/Anlage muss mit seinem Applikations-Programm gewährleisten, dass beim Anlauf oder Wiederanlauf der Steuerung keine gefahrbringenden Bewegungen gestartet werden können.

Ein Wiederanlauf kann z.B. verursacht werden durch:

- Spannungswiederkehr nach Spannungsausfall
- Reset nach Watchdog-Ansprechen wegen zu langer Zykluszeit

© ifm electronic gmbh

2.2 Welche Vorkenntnisse sind notwendig?

215

Das Dokument richtet sich an Personen, die über Kenntnisse der Steuerungstechnik und SPS-Programmierkenntnisse mit IEC 61131-3 verfügen.

Wenn dieses Gerät über eine SPS verfügt, sollten die Personen zusätzlich mit der Software CoDeSys vertraut sein.

Das Dokument richtet sich an Fachkräfte. Dabei handelt es sich um Personen, die aufgrund ihrer einschlägigen Ausbildung und ihrer Erfahrung befähigt sind, Risiken zu erkennen und mögliche Gefährdungen zu vermeiden, die der Betrieb oder die Instandhaltung eines Produkts verursachen kann. Das Dokument enthält Angaben zum korrekten Umgang mit dem Produkt.

Lesen Sie dieses Dokument vor dem Einsatz, damit Sie mit Einsatzbedingungen, Installation und Betrieb vertraut werden. Bewahren Sie das Dokument während der gesamten Einsatzdauer des Gerätes auf.

Befolgen Sie die Sicherheitshinweise.

© ifm electronic gmbh

3 Systembeschreibung

Inhalt:

Angaben zum Gerät 9
Angaben zur Software (Linux | CoDeSys v2.3)....9
Steuerungskonfiguration 10

975

3.1 Angaben zum Gerät

1316

Diese Anleitung beschreibt die PDM360-Monitor-Gerätefamilie der **ifm electronic gmbh** mit Linux-Betriebssystem für den mobilen Einsatz:

- PDM360compact: CR1052, CR1053, CR1055, CR1056

3.2 Angaben zur Software (Linux | CoDeSys v2.3)

3054

Wir beziehen uns in dieser Anleitung auf CoDeSys ab Version 2.3.

Im "Programmierhandbuch CoDeSys 2.3" erhalten Sie weitergehende Informationen über die Nutzung des Programmiersystems "CoDeSys for Automation Alliance". Dieses Handbuch steht auf der **ifm**-Internetseite als kostenloser Download zur Verfügung:
→ www.ifm.com > Land wählen > [Service] > [Download] > [Steuerungssysteme]
→ **ecomat*mobile***-DVD "Software, tools and documentation"

Die Applikations-Software nach IEC 61131-3 kann vom Anwender komfortabel mit dem Programmiersystem CoDeSys selbst erstellt werden. Für den Einsatz dieser Software auf dem PC gelten folgende Mindest-Systemvoraussetzungen:

- CPU Pentium II, 500 MHz
- Arbeitsspeicher (RAM) 128 MB, empfohlen: 256 MB
- Freier Festplattenspeicher (HD) 100 MB
- Betriebssystem Windows 2000 oder höher
- CD-ROM-Laufwerk

Weitere Details zur aktuellen CoDeSys-Software:
DE: → http://www.3s-software.com/index.shtml?de_oem1
UK: → http://www.3s-software.com/index.shtml?en_oem1
FR: → http://www.3s-software.com/index.shtml?fr_oem1

Der Anwender muss außerdem beachten, welcher Softwarestand (speziell beim R360-Betriebssystem und den Funktionsbibliotheken) zum Einsatz kommt.

© ifm electronic gmbh

HINWEIS

Es müssen immer die zum gewählten Target passenden Software-Stände zum Einsatz kommen:
- der Steuerungskonfiguration (`ifm_CRnnnn_Vxxyyyzz.CFG`),
- der Gerätebibliothek (`ifm_CRnnnn_Vxxyyzz.LIB`) und
- der weiteren Dateien (→ Kapitel *Übersicht der verwendeten Dateien und Bibliotheken*, → Seite 322).

CRnnnn	Geräte-Artikelnummer
Vxx: 00...99	Versionsnummer
yy: 00...99	Release-Nummer
zz: 00...99	Patch-Nummer

Dabei müssen der Basisdateiname (z.B. "`CR1051`") und die Software-Versionsnummer "xx" (z.B. "`04`") überall den gleichen Wert haben! Andernfalls geht das Gerät in den STOP-Zustand.

Die Werte für "yy" (Release-Nummer) und "zz" (Patch-Nummer) müssen **nicht** übereinstimmen.

Folgende Dateien müssen ebenfalls geladen sein:
- die zum Projekt erforderlichen internen Bibliotheken (in IEC 1131 erstellt),
- die Konfigurationsdateien (`*.CFG`)
- und die Target-Dateien (`*.TRG`).

Es kann vorkommen, dass das Zielsystem mit Ihrer aktuell installierten Version von CoDeSys nicht oder nur teilweise programmiert werden kann. Im diesem Fall wenden Sie sich bitte an den technischen Support der **ifm electronic gmbh**.

WARNUNG

Für die sichere Funktion der Applikations-Programme, die vom Anwender erstellt werden, ist dieser selbst verantwortlich. Bei Bedarf muss er zusätzlich entsprechend der nationalen Vorschriften eine Abnahme durch entsprechende Prüf- und Überwachungsorganisationen durchführen lassen.

Dieses Gerät enthält Open Source Software entsprechend der GPL:
→ http://www.gnu.org/licenses

3.3 Steuerungskonfiguration

1797

Bei dem Steuerungssystem **ecomat*mobile*** handelt es sich um ein Gerätekonzept für den Serieneinsatz. Das bedeutet, dass die Geräte optimal auf den jeweiligen Einsatzfall konfiguriert werden können.

Die aktuelle Version der **ecomat*mobile***-Software kann über www.ifm.com aus dem Internet geladen werden.
→ *Target einrichten*, → Seite 39.

Ob bestimmte in der Dokumentation beschriebene Funktionen, Hardwareoptionen, Ein- und Ausgänge in der betreffenden Hardware verfügbar sind, muss in jedem Fall vor Einsatz der Geräte überprüft werden.

© ifm electronic gmbh

4 Konfigurationen

Inhalt:

Geräteparameter einstellen (Setup) 11
Programmierschnittstellen 28
Zeichensätze (Fonts) installieren 36
Programmiersystem einrichten 39
Erste Schritte 57
Geräte-Update auf neue Software-Version 63

3615

4.1 Geräteparameter einstellen (Setup)

Inhalt:

Setup starten 12
Ethernet-Schnittstelle: IP-Adresse einstellen 13
Ethernet-Schnittstelle: Subnetzmaske einstellen 15
Startbild des PDM ändern 17
Echtzeituhr (RTC) einstellen 20
Funktion von Tasten, Drehgeber und LEDs prüfen 21
CAN-Schnittstelle prüfen 22
CoDeSys-Kommunikationsparameter einstellen 24
PDM-Setup verlassen, Gerät neu starten 27

7306

In diesem Abschnitt erfahren Sie, wie Sie das Gerät mit dem internen Geräte-Setup einstellen können.

Darstellung und Funktionsmöglichkeiten des Setups sind vom Gerät abhängig und können bei kundenspezifschen Geräten von der in dieser Anleitung gezeigten Version abweichen.

© ifm electronic gmbh

4.1.1 Setup starten

9796

So erreichen Sie das Setup-Menü:

- Wenn im PDM kein gültiges Projekt gespeichert ist:
  Das PDM öffnet nach dem Geräte-Neustart automatisch das Setup-Menü.
- Wenn im PDM ein gültiges Projekt gespeichert ist:
  Beim Einschalten der Versorgungsspannung ca. 1 Sekunde lang die Tastenkombination [F1 esc]+[F2] betätigen.

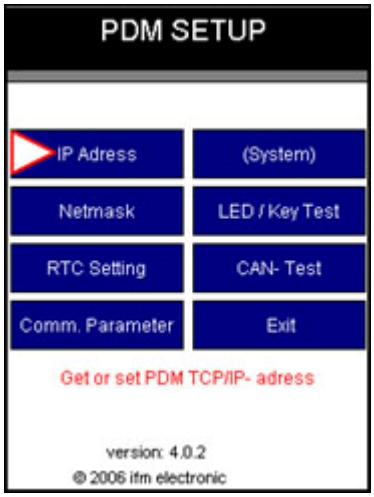


Bild: Setup-Startseite

> Ein rot-weißer Pfeil zeigt auf den gewählten Menüpunkt.

> In der weißen Fußzeile erscheint eine kurze Beschreibung des gewählten Menüpunkts.

► Mit gewünschten Setup-Menüpunkt wählen und mit aktivieren.

> Das PDM wechselt auf das gewählte Setup-Menü.

Bedeutung der Setup-Menüpunkte:

Setup-Feld	Bedeutung
IP Address	Aktuelle IP-Adresse der Ethernet-Schnittstelle lesen oder verändern → *Ethernet-Schnittstelle: IP-Adresse einstellen,* → Seite 13
(System)	Startbild ändern → *Startbild des PDM ändern*, → Seite 17
Netmask	Aktuelle Subnetzmaske der Ethernet-Schnittstelle lesen oder verändern → *Ethernet-Schnittstelle: Subnetzmaske einstellen*, → Seite 15
LED / Key Test	Drehgeber, Tasten und LEDs testen → *Funktion von Tasten, Drehgeber und LEDs prüfen*, → Seite 21
RTC Setting	Echtzeituhr lesen oder verändern → *Echtzeituhr (RTC) einstellen*, → Seite 20
CAN-Test	Testen der CAN-Schnittstelle via Senden oder Empfangen von Datenpaketen → *CAN-Schnittstelle prüfen*, → Seite 22
Comm. Parameter	Baudrate der seriellen Schnittstelle lesen oder verändern → *Serielle Schnittstelle einstellen*, → Seite 25 Node-ID der CAN-Schnittstelle lesen oder verändern → *CAN Download-ID einstellen*, → Seite 26
Exit	Setup verlassen und PDM neu starten → *PDM-Setup verlassen, Gerät neu starten*, → Seite 27

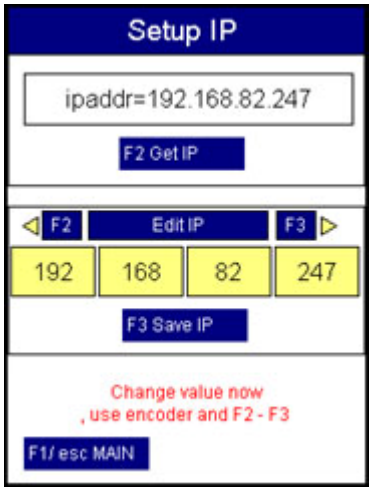
Darstellung und Funktionsmöglichkeiten des Setups sind vom Gerät abhängig und können bei kundenspezifschen Geräten von der in dieser Anleitung gezeigten Version abweichen.

© ifm electronic gmbh

4.1.2 Ethernet-Schnittstelle: IP-Adresse einstellen

9798

► Im Setup-Startbild mit den Menüpunkt [IP Address] wählen und mit aktivieren.



> Das Menübild [Setup IP] erscheint:

► Mit [F2] "Get IP" die aktuelle IP-Adresse lesen.

> Im Feld [ipaddr=] erscheint der aktuell gültige Wert.

> Im Feld [Edit IP] erscheint der vom Hersteller voreingestellte Wert.

► Mit den Zahlenblock wählen (wird gerahmt dargestellt), dessen Wert geändert werden soll, und mit aktivieren.

> Im aktiven Zahlenblock erscheint ein Cursor vor der linken Ziffer.

► Mit [F2]/[F3] "Edit IP" den Cursor vor die zu ändernde Ziffer stellen.

► Mit Ziffer im Bereich 0...9 ändern.

► Bei Bedarf mit [F2]/[F3] "Edit IP" weitere Ziffer des Zahlenblocks wählen und mit ändern.
Zulässige Werte: 000...255 für die linken 3 Zahlenblocks, 001...254 für den rechten Zahlenblock.
Folgende IP-Adressen sind für netzinterne Zwecke reserviert und deshalb als Teilnehmer-Adresse nicht zulässig: nnn.nnn.nnn.0 | nnn.nnn.nnn.255.

► Mit Eingabe-Modus verlassen (nur möglich bei gültigen Werten).

> Zahlenblock wird wieder normal.

► Mit anderen zu ändernden Zahlenblöcken genauso verfahren.

► Mit [F3] "Save IP" die neue IP-Adresse speichern.

► Nach Fertig-Meldung mit [F2] "Get IP" die gültige Adresse in die Anzeige laden.

► Mit [F1 esc] "MAIN" zurück zum Setup-Menü.

► Auch die Subnetzmaske überprüfen und bei Bedarf einstellen!
→ *Ethernet-Schnittstelle: Subnetzmaske einstellen*, → Seite 15

© ifm electronic gmbh

HINWEIS

Im Ethernet-Netzwerk MUSS jede IP-Adresse einmalig sein.

Folgende IP-Adressen sind für netzinterne Zwecke reserviert und deshalb als Teilnehmer-Adresse nicht zulässig: nnn.nnn.nnn.0 | nnn.nnn.nnn.255.

Nur Netzwerkteilnehmer, deren Subnetzmaske identisch ist und deren IP-Adressen im Rahmen der Subnetzmaske identisch sind, können miteinander kommunizieren.

Regel:
Bei Teil-Subnetzmaske = 255 müssen die Teil-IP-Adressen identisch sein.
Bei Teil-Subnetzmaske = 0 müssen sich die Teil-IP-Adressen unterscheiden

Wenn Subnetzmaske = 255.255.255.0, dann sind 254 miteinander kommunizierende Teilnehmer im Netz möglich.
Wenn Subnetzmaske = 255.255.0.0, dann sind 256x254 = 65 024 miteinander kommunizierende Teilnehmer im Netz möglich.

Im selben physikalischen Netzwerk sind unterschiedliche Subnetzmasken der Teilnehmer zulässig. Sie bilden unterschiedliche Teilnehmergruppen, die nicht mit Teilnehmergruppen mit anderen Subnetzmasken kommunizieren können.

Im Zweifelsfall oder bei Problemen fragen Sie bitte Ihren System-Administrator.

Beispiele:

Teilnehmer A IP-Adresse	Teilnehmer A Subnetzmaske	Teilnehmer B IP-Adresse	Teilnehmer B Subnetzmaske	Kommunikation der Teilnehmer möglich?
192.168.82.247	255.255.255.0	192.168.82.10	255.255.255.0	ja, 254 Teilnehmer möglich
192.168.82.**247**	255.255.255.0	192.168.82.**247**	255.255.255.0	nein (gleiche IP-Adresse)
192.168.82.247	255.255.**255**.0	192.168.82.10	255.255.**0**.0	nein (unterschiedliche Subnetzmaske)
192.168.**82**.247	255.255.255.0	192.168.**116**.10	255.255.255.0	nein (unterschiedlicher IP-Adressbereich: 82 vs. 116)
192.168.222.213	255.255.0.0	192.168.222.123	255.255.0.0	ja, 65 024 Teilnehmer möglich
192.168.111.213	255.255.0.0	192.168.222.123	255.255.0.0	ja, 65 024 Teilnehmer möglich
192.168.82.247	255.255.255.0	192.168.82.**0**	255.255.255.0	nein; gesamtes Netz gestört, weil IP-Adresse xxx.xxx.xxx.0 nicht zulässig

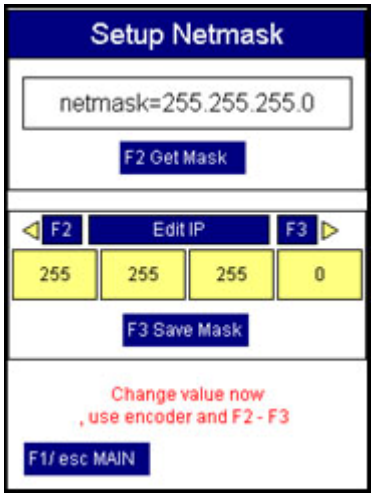
Geänderte Werte sind erst nach einem Neustart des Geräts (z.B. Spannungsversorgung AUS / EIN) aktiv.

© ifm electronic gmbh

4.1.3 Ethernet-Schnittstelle: Subnetzmaske einstellen

9800

► Im Setup-Startbild mit den Menüpunkt [Netmask] wählen und mit aktivieren.



> Das Menübild [Setup Netmask] erscheint:

► Mit [F2] "Get Mask" die aktuelle IP-Adresse lesen.

> Im Feld [netmask=] erscheint der aktuell gültige Wert.

> Im Feld [Edit IP] erscheint der vom Hersteller voreingestellte Wert.

► Mit den Zahlenblock wählen (wird gerahmt dargestellt), dessen Wert geändert werden soll, und mit aktivieren.

> Im aktiven Zahlenblock erscheint ein Cursor vor der linken Ziffer.

► Mit [F2]/[F3] "Edit IP" den Cursor vor die zu ändernde Ziffer stellen.

► Mit Ziffer im Bereich 0...9 ändern.

► Bei Bedarf mit [F2]/[F3] "Edit IP" weitere Ziffer des Zahlenblocks wählen und mit ändern. Zulässige Werte: 000...255; sinnvolle Werte: 000 oder 255.

► Mit Eingabe-Modus verlassen (nur möglich bei gültigen Werten).

> Zahlenblock wird wieder normal.

► Mit anderen zu ändernden Zahlenblöcken genauso verfahren.

► Mit [F3] "Save Mask" die neue Subnetzmaske speichern.

► Nach Fertig-Meldung mit [F2] "Get Mask" die gültige Subnetzmaske in die Anzeige laden.

► Mit [F1 esc] "MAIN" zurück zum Setup-Menü.

HINWEIS

Im Ethernet-Netzwerk MUSS jede IP-Adresse einmalig sein.

Folgende IP-Adressen sind für netzinterne Zwecke reserviert und deshalb als Teilnehmer-Adresse nicht zulässig: nnn.nnn.nnn.0 | nnn.nnn.nnn.255.

Nur Netzwerkteilnehmer, deren Subnetzmaske identisch ist und deren IP-Adressen im Rahmen der Subnetzmaske identisch sind, können miteinander kommunizieren.

Regel:
Bei Teil-Subnetzmaske = 255 müssen die Teil-IP-Adressen identisch sein.
Bei Teil-Subnetzmaske = 0 müssen sich die Teil-IP-Adressen unterscheiden

Wenn Subnetzmaske = 255.255.255.0, dann sind 254 miteinander kommunizierende Teilnehmer im Netz möglich.
Wenn Subnetzmaske = 255.255.0.0, dann sind 256x254 = 65 024 miteinander kommunizierende Teilnehmer im Netz möglich.

Im selben physikalischen Netzwerk sind unterschiedliche Subnetzmasken der Teilnehmer zulässig. Sie bilden unterschiedliche Teilnehmergruppen, die nicht mit Teilnehmergruppen mit anderen Subnetzmasken kommunizieren können.

Im Zweifelsfall oder bei Problemen fragen Sie bitte Ihren System-Administrator.

© ifm electronic gmbh

Beispiele:

Teilnehmer A IP-Adresse	Teilnehmer A Subnetzmaske	Teilnehmer B IP-Adresse	Teilnehmer B Subnetzmaske	Kommunikation der Teilnehmer möglich?
192.168.82.247	255.255.255.0	192.168.82.10	255.255.255.0	ja, 254 Teilnehmer möglich
192.168.82.**247**	255.255.255.0	192.168.82.**247**	255.255.255.0	nein (gleiche IP-Adresse)
192.168.82.247	255.255.**255**.0	192.168.82.10	255.255.**0**.0	nein (unterschiedliche Subnetzmaske)
192.168.**82**.247	255.255.255.0	192.168.**116**.10	255.255.255.0	nein (unterschiedlicher IP-Adressbereich: 82 vs. 116)
192.168.222.213	255.255.0.0	192.168.222.123	255.255.0.0	ja, 65 024 Teilnehmer möglich
192.168.111.213	255.255.0.0	192.168.222.123	255.255.0.0	ja, 65 024 Teilnehmer möglich
192.168.82.247	255.255.255.0	192.168.82.**0**	255.255.255.0	nein; gesamtes Netz gestört, weil IP-Adresse xxx.xxx.xxx.0 nicht zulässig

Geänderte Werte sind erst nach einem Neustart des Geräts (z.B. Spannungsversorgung AUS / EIN) aktiv.

© ifm electronic gmbh

4.1.4 Startbild des PDM ändern

3014

Beim Booten des PDM erscheint ein Startbild. Hier beschreiben wir, wie der Kunde ein eigenes Startbild einsetzen kann.

Für Geräte der PDM360smart-Familie (z.B. CR1070, CR1071) ist dies nicht verfügbar.

Bild-Vorgaben:

	CR1051	CR1050	CR1052, CR1053	CR1055, CR1056
Datei-Typ	Bitmap (`*.bmp`)			
Dateiname	nur Kleinbuchstaben, Namenskonvention = 8.3			
Bildgröße	320x240 Pixel	320x240 Pixel	240x320 Pixel	320x240 Pixel
Farben	8 Bit = 256 Farben	1 Bit = nur Schwarz und Weiß, keine Graustufen		
Speicherbedarf	ca. 77 kByte	ca. 10 kByte		

Davon abweichende Bilder akzeptiert das PDM nicht (→ auch Kapitel *Visualisierungen im Gerät*, → Seite 80).

► Auf dem Computer das gewünschte Startbild nach diesen Vorgaben erzeugen.

Bild übertragen per FTP

9805

► Die IP-Adresse und die Subnetzmaske des Computers passend zu den Angaben im PDM einstellen (→ Kapitel *Ethernet-Schnittstelle: IP-Adresse einstellen*, → Seite 13).

Wenn Einstellungen der Firewall eine FTP-Verbindung behindern oder nicht zulassen: Firewall-Einstellung ändern oder Firewall temporär abschalten.

Im Zweifelsfall oder bei Problemen fragen Sie bitte Ihren System-Administrator.

► Im Internet-Browser (oder Freeware-FTP-Software) eingeben: ftp://192.168.82.247
(hier die tatsächliche IP-Adresse des PDM angeben)

> Internet-Browser zeigt die Verzeichnis-Struktur auf dem PDM an, z.B.:

► Rechtsklick in eine freie Stelle dieser Anzeige.

► Im Kontextmenü klicken auf [Anmelden als...].

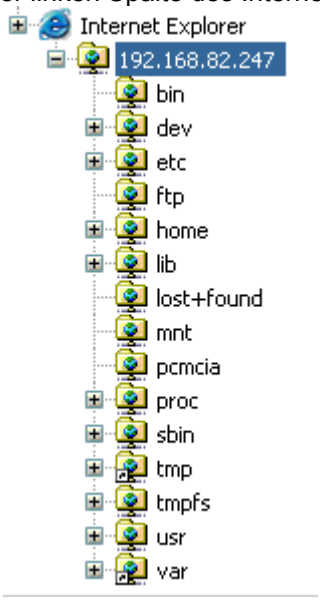
> Anmeldefenster des PDM erscheint.

► Anmeldename = `root`
Kennwort = `root` (Kleinbuchstaben!)
Mit [OK] beim Betriebssystem des PDM anmelden.

► Im Menü klicken auf [Ansicht] > [Explorer-Leiste] > [Ordner].

© leonic gmbh

> In der linken Spalte des Internet-Browser erscheint die Ordnerstruktur des PDM, z.B. (Ausschnitt):



► Die neue Startbild-Datei in der lokalen Ordnerstruktur suchen und markieren.

► Datei mit [Bearbeiten] > [Kopieren] in die Zwischenablage kopieren.

► In der Ordnerstruktur wechseln auf das Verzeichnis `tmp` des PDM.

► Datei mit [Bearbeiten] > [Einfügen] aus der Zwischenablage in das Verzeichnis `tmp` des PDM kopieren.

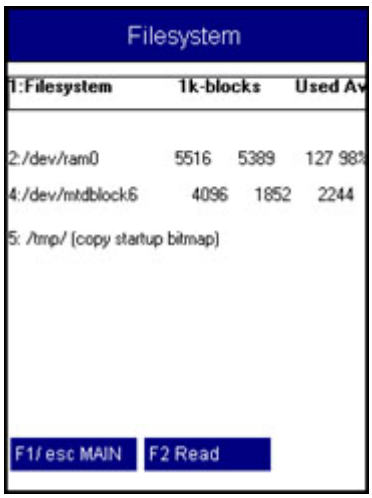
> Der Computer sendet die Datei per FTP-Übertragung zum PDM.

© ifm electronic gmbh

PDM-Setup: Startbild des PDM ändern

9802

► Im Setup-Startbild mit den Menüpunkt [System] wählen und mit aktivieren.



> Das Menübild [Filesystem] erscheint.

► Mit [F2] "Read" die Anzeige aktualisieren.

► Mit das Verzeichnis "`/tmp/`" wählen und mit aktivieren.

Das Dateisystem des Geräts arbeitet auf Linux-Basis.

> Das Menübild [Filesystem /tmp/] erscheint.

► Mit [F2] "Refresh" die Anzeige aktualisieren.

► Mit gewünschte BMP-Datei wählen und mit aktivieren.

> Bei gültiger Datei erscheint Funktionstaste [F3] "copy bmp".

► Mit [F3] "copy bmp" das gewählte Bild als neues Startbild in den Flash-Speicher des PDM verschieben.

► Nach erfolgreicher Übertragung erscheint die Statusmeldung "... bytes copied! Finished".

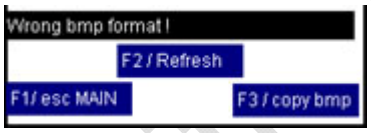


Bild: Beispiel für eine Fehleranzeige

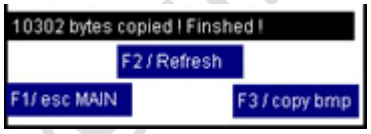


Bild: Beispiel für eine Fertigmeldung nach erfolgreicher Übertragung

► Das Setup-Programm mit [F1 esc] "Main" > [Exit] > [F2] verlassen.

> Beim nächsten Systemstart wird das neue Startbild angezeigt.

© ifm electronic gmbh

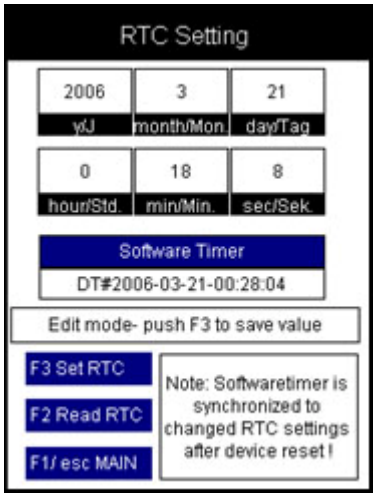
4.1.5 Echtzeituhr (RTC) einstellen

9810

Gilt nur für folgende Geräte:
- PDM360compact: CR1053, CR1056

Die Echzeituhr (**R**eal**T**ime**C**lock) ist für zeitgesteuerte Funktionen in der Applikation vorgesehen. Uhrzeit und Datum werden bereits im Setup-Startbild angezeigt.

► Mit Setup-Menü [RTC Setting] wählen und mit aktivieren.



> Menü [RTC Setting] erscheint:

- Eingabefeld Datum im Format Jahr / Monat / Tag
- Eingabefeld Uhrzeit im Format Stunde / Minute / Sekunde
- Anzeige Software-Zeit (= die Zeit, mit der die Software im CoDeSys-Projekt arbeitet)

► Mit [F2] "Read RTC" erhalten Sie die aktuelle Einstellung der Echtzeituhr.

► Mit den Wert wählen (wird gerahmt dargestellt), der geändert werden soll und mit das Ändern aktivieren.

► Mit den Wert ändern und mit den geänderten Wert speichern.

► Mit anderen zu ändernden Zahlenblöcken genauso verfahren.

► Mit [F3] "Set RTC" die Änderung übernehmen.

> Gerät bootet neu.

> Neues Datum und Uhrzeit sind von der Softwarezeit (für das CoDeSys-Projekt) übernommen.

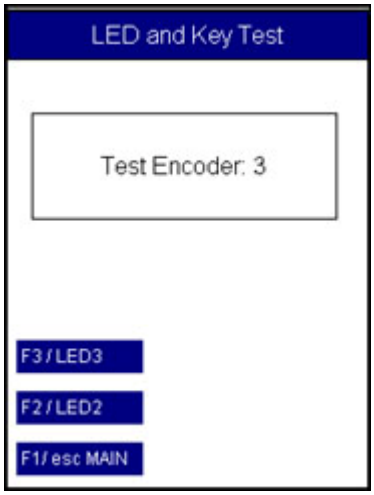
© ifm electronic gmbh

4.1.6 Funktion von Tasten, Drehgeber und LEDs prüfen

9808

Dieser Menüpunkt dient zur Funktionsüberprüfung der Bedienelemente und LEDs des Geräts.

► Im Setup-Startbild mit den Menüpunkt [LED / Key Test] wählen und mit aktivieren.



> Das Menübild [LED and Key Test] erscheint.

► Mit verändern Sie den Wert in [Test Encoder].

► Drücken einer Taste [F2]...[F3] oder des Drehgeber-Drehknopfs aktiviert die zugehörige LED-Tastenbeleuchtung.

► Mit [F1 esc] "MAIN" den Setup-Menüpunkt wieder verlassen.

© ifm electronic gmbh

4.1.7 CAN-Schnittstelle prüfen

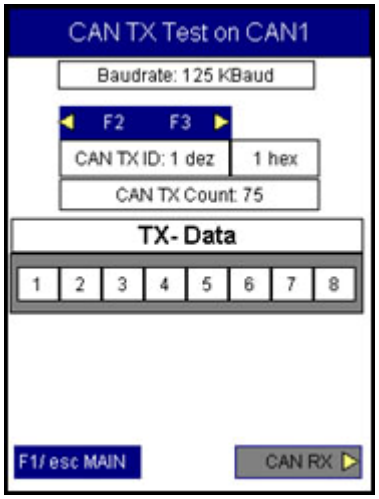
9812

Zum Testen der CAN-Schnittstelle können Sie CAN-Telegramme an andere CAN-Teilnehmer senden oder die CAN-Telegramme anderer CAN-Teilnehmer empfangen.

Test: CAN-Telegramme senden (TX)

9813

► Im Setup-Startbild mit den Menüpunkt [CAN-Test] wählen und mit aktivieren.



> Das Menübild [CAN TX Test on CAN1] erscheint:
- Baudrate 125 kBaud fest eingestellt
- Eingabefeld CAN TX ID (dezimal)
- Anzeige CAN TX ID (hexadezimal)
- Anzeige Anzahl gesendete CAN-Pakete
- 8 Eingabefelder für Sende-Daten

► Mit das Feld wählen (wird gerahmt dargestellt), dessen Wert geändert werden soll, und mit das Ändern aktivieren.

► Im Feld [CAN TX ID] mit [F2]/[F3] den Cursor vor die zu ändernde Ziffer stellen.

► Im Feld CAN TX ID mit Ziffer im Bereich 0...9 ändern.
Bei Bedarf mit [F3]/[F4] "edit pos" weitere Ziffer des Zahlenblocks wählen und mit ändern.
Die CAN TX ID muss im Bereich 0...2048 sein! Dies entspricht einem 11-Bit-Identifier.

► Mit den geänderten Wert speichern.

> Der neue CAN-TX-ID erscheint nun auch als Hexadezimal-Wert.

► In den Feldern TX-Data mit Ziffer im Bereich 0...255 ändern.

► Mit den geänderten Wert speichern.

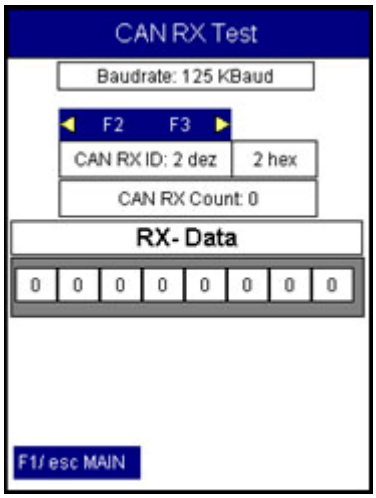
> Gerät sendet die eingestellten Daten als "Broadcast" an alle Teilnehmer.
Mit jedem gesendeten Daten-Paket erhöht sich der Zählerstand in [CAN TX Count] um 1.

© ifm electronic gmbh

Test: CAN-Telegramme empfangen (RX)

9814

► Im Menübild [CAN TX Test on CAN1] mit [F3] "CAN RX" umschalten auf "CAN-Telegramme empfangen".



> Das Menübild [CAN RX Test] erscheint:
- Baudrate 125 kBaud fest eingestellt
- Eingabefeld CAN RX ID (dezimal)
- Anzeige CAN RX ID (hexadezimal)
- Anzeige Anzahl empfangene CAN-Pakete
- 8 Anzeigefelder für Empfangs-Daten

► Mit das Feld [CAN RX ID] wählen (wird gerahmt dargestellt) und mit das Ändern aktivieren.

► Mit [F2]/[F3] den Cursor vor die zu ändernde Ziffer stellen.

► Mit den Wert ändern und mit den geänderten Wert speichern.

> Gerät empfängt mit den eingestellten Daten.
Anzeige der empfangenen Daten in den RX-Data-Feldern.

> Ist ein Empfang nicht möglich, zeigen der Zähler [CAN RX Count] und RX-Data jeweils den Wert 0 an.

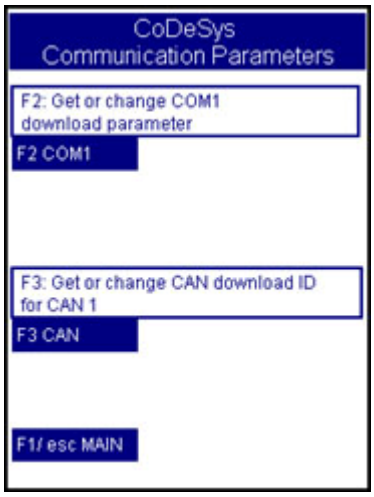
© ifm electronic gmbh

4.1.8 CoDeSys-Kommunikationsparameter einstellen

9817

Hier können Sie die Kommunikationsparameter für den Programm-Download über die serielle Schnittstelle oder über die CAN-Schnittstelle einstellen.

► Im Setup-Startbild mit den Menüpunkt [Comm. Parameter] wählen und mit aktivieren.



> Das Menübild [CoDeSys Communication Parameters] erscheint.

Von hier aus können Sie die Schnittstelle wählen, deren Parameter eingesehen oder verändert werden sollen:

► Mit [F2] die serielle Schnittstelle einstellen.

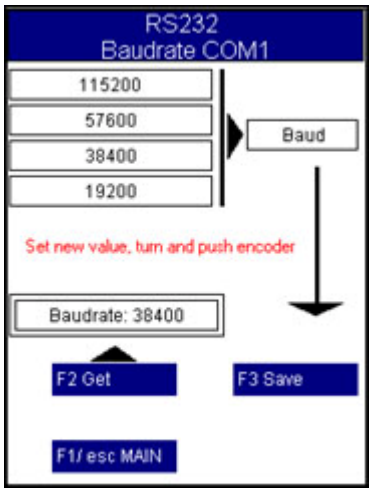
► Mit [F3] die CAN-Schnittstelle einstellen.

© ifm electronic gmbh

Serielle Schnittstelle einstellen

9819

► Im Menübild [CoDeSys Communication Parameters] mit [F2] umschalten auf das Menü [RS232 Baudrate COM1].



> Das Menübild [RS232 Baudrate COM1] erscheint.

► Mit [F2] "Get" den aktuell gültigen Wert anzeigen lassen.

► Mit die gewünschte Baudrate aus der Tabelle wählen (wird gerahmt dargestellt) und mit aktivieren.

► Mit [F3] "Save" den neuen Wert speichern.

> Taste [F2] "Get Baud" leuchtet.

► Mit [F2] "Get Baud" den aktuell gültigen Wert übernehmen.

> Nach einem Neustart (Spannungsversorgung Aus/Ein) arbeitet das Gerät mit den neuen Einstellungen.

© ifm electronic gmbh

CAN Download-ID einstellen

9821

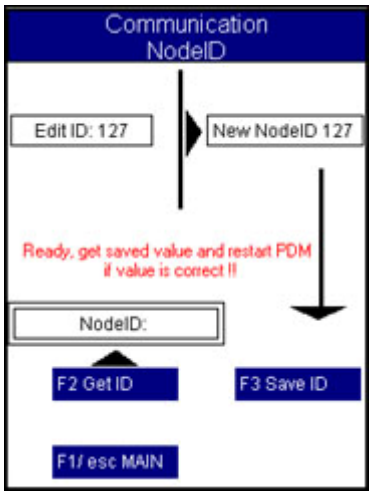
Dieses Gerät unterstützt keinen CAN-Download über CoDeSys!

HINWEIS

Der CAN-Download-ID des Geräts muss mit dem in CoDeSys eingestellten CAN-Download-ID übereinstimmen!

Im CAN-Netzwerk müssen die CAN-Download-IDs einmalig sein!

Im Menübild [CoDeSys Communication Parameters] mit [F3] umschalten auf das Menübild [Communication Node ID].



> Das Menübild [Communication Node ID] erscheint.

► Mit [F2] "Get ID" den aktuell gültigen Wert anzeigen lassen.

► Mit das Feld [Edit ID] wählen (wird gerahmt dargestellt) und mit aktivieren.

> Anzeigefeld wechselt zum Eingabefeld.

► Mit den Wert ändern (1...127) und mit den geänderten Wert speichern.

> Eingabefeld wechselt zum Anzeigefeld.

► Mit [F3] "Save" den neuen Wert speichern.

> Taste [F2] "Get ID" leuchtet.

► Mit [F2] "Get ID" den aktuellen Wert übernehmen.

> Nach einem Neustart (Spannungsversorgung Aus/Ein) arbeitet das Gerät mit den neuen Einstellungen.

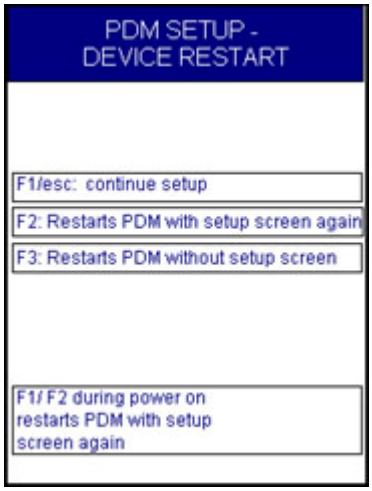
© ifm electronic gmbh

4.1.9 PDM-Setup verlassen, Gerät neu starten

9823

In diesem Menü können Sie wählen, ob und wie Sie das PDM-Setup verlassen wollen.

► Im Setup-Startbild mit das Menübild [Exit] wählen und mit aktivieren.


PDM SETUP -
DEVICE RESTART
F1/esc: continue setup
F2: Restarts PDM with setup screen again
F3: Restarts PDM without setup screen
F1/ F2 during power on
restarts PDM with setup
screen again

> Das Menübild [PDM Setup - Device Restart] erscheint.

Folgende Wahlmöglichkeiten haben Sie:

- Mit [F1/esc] zurück in das Setup-Startbild ohne Geräte-Neustart.
- Mit [F2] das Gerät neu starten.
  Danach wieder das Setup-Startbild öffnen.
- Mit [F3] das Gerät neu starten und (wenn vorhanden) die Applikation starten.
  Ohne Applikation erscheint wieder das Setup-Startbild.

Grundsätzlich können Sie bei jedem Geräte-Neustart mit der Tastenkombination [F1 esc]+[F2] (ca. 1 Sekunde lang betätigen) in das Setup-Menü gelangen.

© ifm electronic gmbh

4.2 Programmierschnittstellen

Inhalt:

Programmierung über die Ethernet-Schnittstelle 29
Programmierung über die serielle Schnittstelle RS232 35

3025

Zur Programmierung stehen im PDM derzeit 3 unterschiedliche Schnittstellen zur Verfügung:

- Programmierung über die Ethernet-Schnittstelle,
- Programmierung über die serielle Schnittstelle RS232,
- Programmierung über die CAN-Schnittstelle
  (NICHT für folgende Geräte verfügbar: PDM360compact: CR1052, CR1053, CR1055, CR1056).

© ifm electronic gmbh

4.2.1 Programmierung über die Ethernet-Schnittstelle

Inhalt:

Internet-Protokoll auf dem Computer einstellen 30
Ethernet-Verbindung testen 31
CoDeSys-Kommunikationsparameter für die Ethernet-Schnittstelle einstellen 34

9826

Die Ethernet-Schnittstelle finden Sie auf der Rückseite am Steckanschluss 4.
(Technische Details → Datenblatt)

Die Nutzung der Ethernet-Schnittstelle ist während der Programmentwicklung zu empfehlen, da die Übertragungszeiten besonders kurz sind.

Im Computer wird ebenfalls eine Ethernet-Schnittstelle benötigt.

Verbindung von Computer und Gerät über ein vorhandenes Netzwerk (LAN):
8-poliges Verbindungskabel, M12-Stecker – RJ45-Stecker:

Länge	Artikel-Nr.	Foto
2 m	E11898	
10 m	E12204	
20 m	E12205	

Für den Anschluss im Netzwerk sollte ein erfahrener Anwender oder ein System-Administrator die Einrichtung der Netzwerkadressen und die Konfiguration vornehmen.

Zur Kommunikation zwischen PDM und Computer die IP-Adresse und die Subnetzmaske des PDM beachten!
Werkseinstellung IP-Adresse: 192.168.082.247
Werkseinstellung Subnetzmaske: 255.255.255.0

Wenn Ihr vorhandenes Netzwerk diese Einstellung nicht zulässt, müssen Sie die IP-Adresse und die Subnetzmaske des PDM anpassen:
→ *Ethernet-Schnittstelle: IP-Adresse einstellen*, → Seite 13
→ *Ethernet-Schnittstelle: Subnetzmaske einstellen*, → Seite 15

HINWEIS

Im Ethernet-Netzwerk MUSS jede IP-Adresse einmalig sein.

Folgende IP-Adressen sind für netzinterne Zwecke reserviert und deshalb als Teilnehmer-Adresse nicht zulässig: nnn.nnn.nnn.0 | nnn.nnn.nnn.255.

Nur Netzwerkteilnehmer, deren Subnetzmaske identisch ist und deren IP-Adressen im Rahmen der Subnetzmaske identisch sind, können miteinander kommunizieren.

Regel:
Bei Teil-Subnetzmaske = 255 müssen die Teil-IP-Adressen identisch sein.
Bei Teil-Subnetzmaske = 0 müssen sich die Teil-IP-Adressen unterscheiden

Wenn Subnetzmaske = 255.255.255.0, dann sind 254 miteinander kommunizierende Teilnehmer im Netz möglich.
Wenn Subnetzmaske = 255.255.0.0, dann sind 256x254 = 65 024 miteinander kommunizierende Teilnehmer im Netz möglich.

Im selben physikalischen Netzwerk sind unterschiedliche Subnetzmasken der Teilnehmer zulässig. Sie bilden unterschiedliche Teilnehmergruppen, die nicht mit Teilnehmergruppen mit anderen Subnetzmasken kommunizieren können.

© ifm electronic gmbh

Im Zweifelsfall oder bei Problemen fragen Sie bitte Ihren System-Administrator.

Beispiele:

Teilnehmer A IP-Adresse	Teilnehmer A Subnetzmaske	Teilnehmer B IP-Adresse	Teilnehmer B Subnetzmaske	Kommunikation der Teilnehmer möglich?
192.168.82.247	255.255.255.0	192.168.82.10	255.255.255.0	ja, 254 Teilnehmer möglich
192.168.82.**247**	255.255.255.0	192.168.82.**247**	255.255.255.0	nein (gleiche IP-Adresse)
192.168.82.247	255.255.**255**.0	192.168.82.10	255.255.**0**.0	nein (unterschiedliche Subnetzmaske)
192.168.**82**.247	255.255.255.0	192.168.**116**.10	255.255.255.0	nein (unterschiedlicher IP-Adressbereich: 82 vs. 116)
192.168.222.213	255.255.0.0	192.168.222.123	255.255.0.0	ja, 65 024 Teilnehmer möglich
192.168.111.213	255.255.0.0	192.168.222.123	255.255.0.0	ja, 65 024 Teilnehmer möglich
192.168.82.247	255.255.255.0	192.168.82.**0**	255.255.255.0	nein; gesamtes Netz gestört, weil IP-Adresse xxx.xxx.xxx.0 nicht zulässig

Internet-Protokoll auf dem Computer einstellen

3029

Sie müssen vom Administrator berechtigt worden sein, diese Einstellungen vorzunehmen.
Im Zweifelsfall oder bei Problemen fragen Sie bitte Ihren System-Administrator.

► In der Taskleiste Klick auf [Start], dann [Systemsteuerung] > [Netzwerkverbindungen]

> Das Fenster "Netzwerkverbindungen" zeigt eine Liste der installierten Netzwerkkomponenten.

► Rechtsklick auf [LAN-Verbindung].

► Klick auf [Eigenschaften].

> Das Fenster "Eigenschaften von LAN-Verbindung" erscheint.

► [Internetprotokoll (TCP/IP)] markieren.

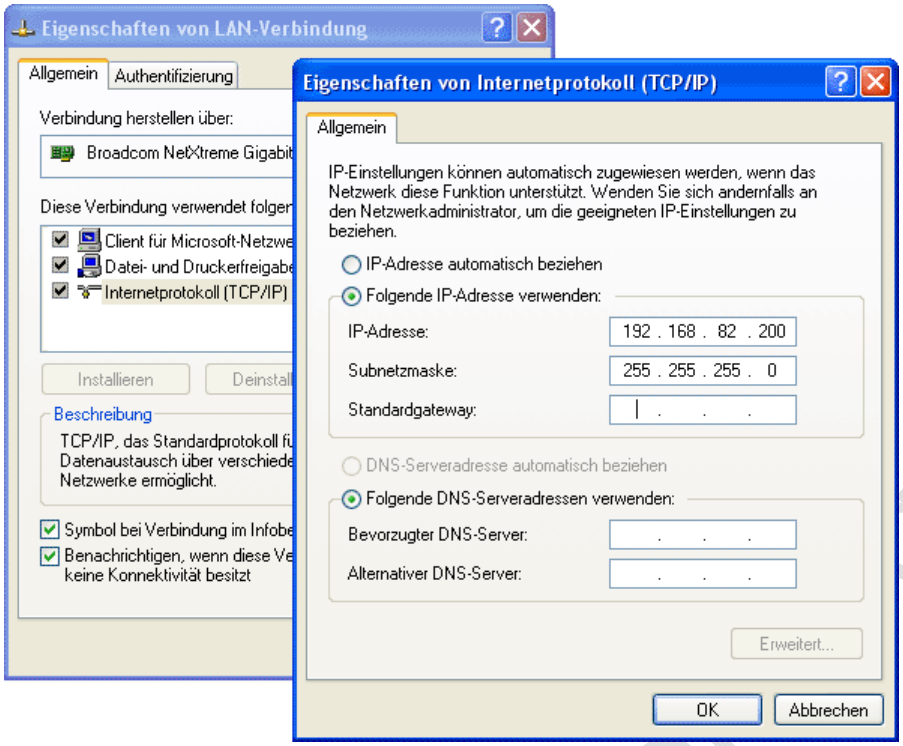
► Klick auf [Eigenschaften].

> Das Fenster "Eigenschaften von Internetprotokoll (TCP/IP)" erscheint.

► Das Optionsfeld [Folgende IP-Adresse verwenden] aktivieren.

© ifm electronic gmbh

► Im Feld [IP-Adresse] z.B. 192.168.82.200 eintragen und im Feld [Subnetzmaske] 255.255.255.0 eintragen (→ Bild).



► Nach [OK] sollten Computer und PDM miteinander kommunizieren können.

Ethernet-Verbindung testen

3030

Nachdem an PDM und Computer die IP-Adressen und die Subnetzmaske eingestellt sind, kann jetzt die Verbindung zwischen Ihrem Computer und dem PDM hergestellt und getestet werden.

► Korrekte Kabelverbindung zwischen beiden Geräten prüfen:

> OK, wenn die orangefarbene LED neben dem Netzwerkstecker leuchtet oder flackert.

Ethernet testen über PING-Funktion

3050

► Mit [Start] > [Alle Programme] > [Zubehör] > [Eingabeaufforderung] die Windows-Kommandozeile aufrufen.

► Hinter der Eingabeaufforderung "`C:\>`" eintippen: `ping 192.168.082.247` [Enter]
Hier die tatsächliche IP-Adresse des PDM angeben.

> Das Betriebssystem des Computer versucht nun 4-mal in Folge, ein 32 Bytes langes Testdaten-Paket zum Gerät zu senden.

> Wenn Verbindung in Ordnung ist, erscheint 4-mal z.B. die folgende Meldung:
`Antwort von 192.168.82.247: Bytes=32 Zeit=1ms TTL=252`
gefolgt von einer Statistik. Im Netzwerk kann die Zeitangabe variieren.

> Wenn keine Verbindung zustande gekommen ist, erscheint 4-mal die folgende Meldung:
`Zeitüberschreitung der Anforderung.`
gefolgt von einer Statistik.

► Eingabeaufforderung mit der Eingabe von `exit` [Enter] wieder schließen.

© ifm electronic gmbh

Ethernet testen über Hyper Terminal

3051

► Zu Testzwecken ein Terminalprogramm, welches das TCP/IP-Protokoll unterstützt, starten:
z.B. [Start] > [Alle Programme] > [Zubehör] > [Kommunikation] > [Hyper Terminal]

Info

Üblicherweise ist ein Terminalprogramm, wie z.B. Telnet oder Hyperterminal, auf jeder Windows-Installation vorhanden. Falls dies nicht der Fall ist, müssen Sie das Programm nachinstallieren:

► Windows-CD starten und Menüpunkt [Software] wählen.

► Doppelklick auf [Verbindungen].

> Auswahlmenü erscheint.

► [Hyper Terminal] markieren.

► Einstellungen übernehmen und das Programm installieren.

Im Zweifelsfall oder bei Problemen fragen Sie bitte Ihren System-Administrator.

Im Terminalprogramm müssen Sie diverse Einstellungen vornehmen:

► Als Name der Verbindung `'PDM360'` eintragen.

► Symbol dazu wählen (z.B. → Bild):

► Mit [OK] übernehmen.

> Fenster "Verbinden mit" erscheint:

ifm electronic gmbh

► Verbindung herstellen über = [TCP/IP (Winsock)] wählen.

> Eingabefelder für TCP/IP erscheinen.

► Hostadresse = (z.B.) 192.168.082.247
(tatsächliche IP-Adresse des PDM eintragen, → Bild oben)

► Mit [OK] die Einstellungen bestätigen.

> PDM wird nun erkannt, Login-Kommando erscheint.
Wenn das PDM noch nicht erkannt wird, kann es helfen, sowohl das Terminalprogramm wie auch das PDM neu zu starten.

► Nach [login:] den Begriff `root` eingeben, gefolgt von [Enter].

► Nach [Password:] ebenfalls den Begriff `root` (in Kleinbuchstaben!) eingeben, gefolgt von [Enter].

> Es erscheint eine neue Meldung über den Betriebszustand des PDM, beginnend mit "`BusyBox`".

Damit war Ihr Test erfolgreich. Das Terminalprogramm wird jetzt nicht mehr benötigt.

► Im Terminalprogramm mit Eingeben von `exit` hinter dem "#" und Taste [Enter] die Verbindung beenden.

> In der Statuszeile des Terminalprogramms erscheint die Meldung "Verbindung getrennt".

► Bei Bedarf die Einstellungen mit [Datei] > [speichern] sichern.

> Verbindungs-Einstellungen werden z.B. als `PDM360.ht` gespeichert.

► Mit [Alt]+[F4] das Terminalprogramm beenden.

© ifm electronic gmbh

CoDeSys-Kommunikationsparameter für die Ethernet-Schnittstelle einstellen

3075

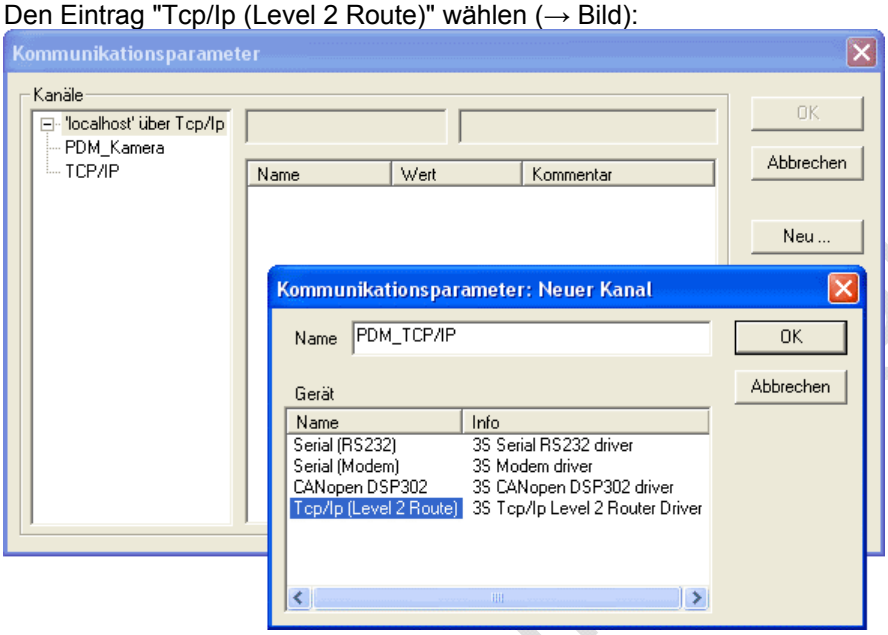
► In CoDeSys [Online] > [Kommunikationsparameter...] klicken.

► Klicken auf [Neu...]

► Fenster "Kommunikationsparameter: Neuer Kanal" erscheint.

► Einen selbsterklärenden Namen vergeben, z.B. "PDM_TCP/IP".

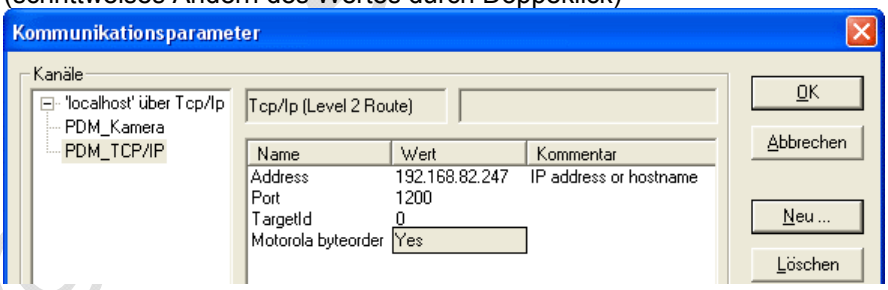
► Den Eintrag "Tcp/Ip (Level 2 Route)" wählen (→ Bild):



► Neue Parameter mit [OK] übernehmen.

► Für den neuen Kanal die folgenden Kommunikationsparameter eintragen (→ Bild unten):
- [Address] = z.B. 192.168.82.247 (hier die tatsächliche IP-Adresse des Geräts eintragen)
- [Motorola byteorder] = Yes (für alle PDM, außer CR107n)
- [Motorola byteorder] = No (für alle Controller und CR107n)

(schrittweises Ändern des Wertes durch Doppeklick)



► Kommunikationsparameter mit [OK] übernehmen.

> Nun sollten CoDeSys und das Gerät über die Ethernet-Schnittstelle kommunizieren können.

4.2.2 Programmierung über die serielle Schnittstelle RS232

9827

Auf der Geräte-Rückseite auf Steckanschluss 2 gibt es 1 serielle Schnittstelle (technische Details → Datenblatt).

Über ein Nullmodemkabel (gekreuzte Datenleitungen) kann die Verbindung zwischen PDM und der seriellen Schnittstelle am Computer hergestellt werden.

CoDeSys-Kommunikationsparameter für die serielle Schnittstelle einstellen

3074

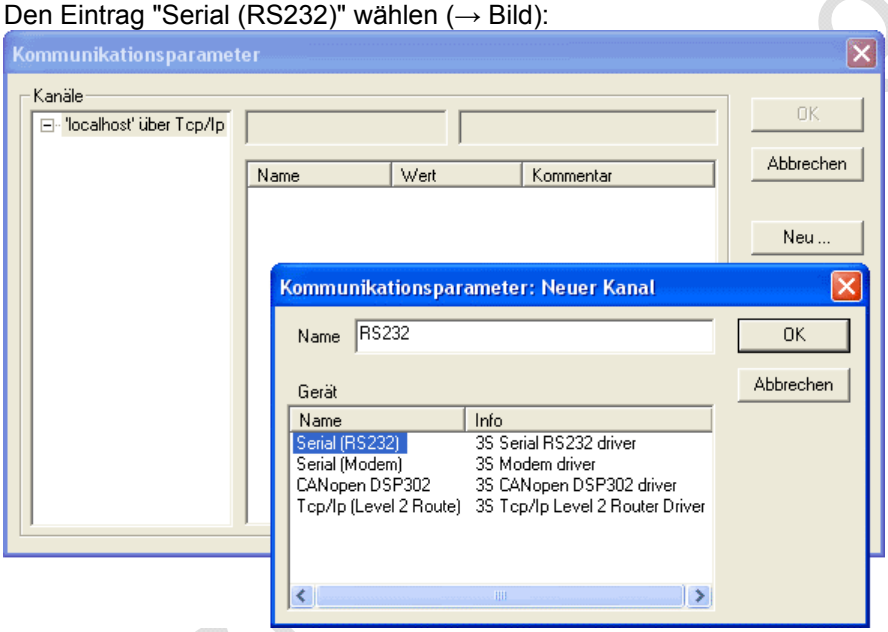
► In CoDeSys [Online] > [Kommunikationsparameter...] klicken.

► Klicken auf [Neu...]

► Fenster "Kommunikationsparameter: Neuer Kanal" erscheint.

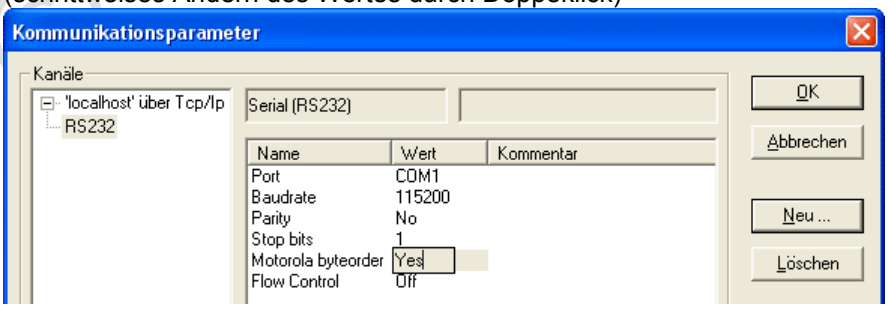
► Einen selbsterklärenden Namen vergeben, z.B. "ifm_RS232".

► Den Eintrag "Serial (RS232)" wählen (→ Bild):



► Für den neuen Kanal die folgenden Kommunikationsparameter eintragen (→ Bild):
- [Baudrate] = 115200
- [Motorola byteorder] = Yes (für alle PDM, außer CR107n)
- [Motorola byteorder] = No (für alle Controller und CR107n)

(schrittweises Ändern des Wertes durch Doppeklick)



► Kommunikationsparameter mit [OK] übernehmen.

> Nun sollten CoDeSys und das Gerät über die serielle Schnittstelle kommunizieren können.

4.3 Zeichensätze (Fonts) installieren

3690

Nicht alle Applikationen kommen mit den werkseitig gelieferten Zeichensätzen (Schriftarten, Fonts) aus. Hier zeigen wir Ihnen am Beispiel "Kyrillisch" einen Weg, weitere Zeichensätze zu ergänzen.

► Auf dem PC die neue Schriftart installieren unter:
[Systemsteuerung] > [Schriftarten] > [Datei] > [Neue Schriftart installieren...].

► PC neu starten.

► Mit einem FTP-Programm die neuen Schriftarten auf das PDM in das Verzeichnis `/home/fonts` kopieren.

► Alle Font-Dateien müssen das Dateiattribut "ausführbar" erhalten. Das kann normalerweise auch mit den FTP-Programmen durchgeführt werden.

► Starten Sie ein Terminalprogramm, um die neuen Font-Dateien verlinken zu können.

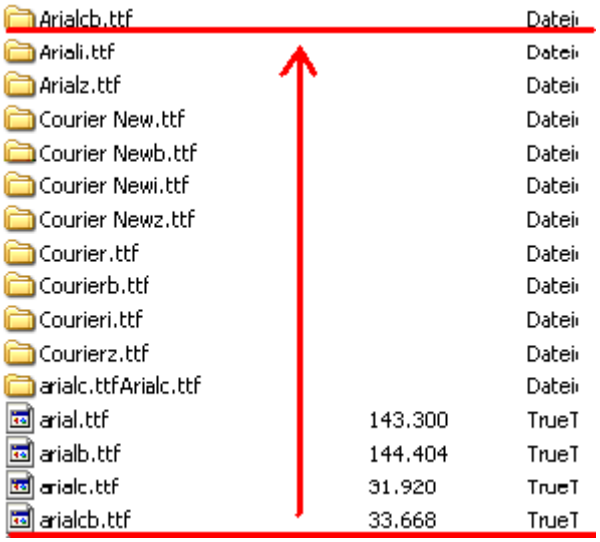
Beispiel:

Für eine neue Font-Datei `arialc.ttf` folgende Kommandozeile eingeben:

```
ln -s arialcb.ttf  “Arialcb.ttf“[ENTER]:
# cd home/fonts
# ls
Arial.ttf             Courier.ttf           arialci.ttf
Arialb.ttf            Courierb.ttf          arialcz.ttf
Arialc.ttf            Courieri.ttf          ariali.ttf
Ariali.ttf            Courierz.ttf          arialz.ttf
Arialz.ttf            arial.ttf             courier.ttf
Courier New.ttf       arialb.ttf            courierb.ttf
Courier Newb.ttf      arialc.ttf            courieri.ttf
Courier Newi.ttf      arialc.ttfArialc.ttf  courierz.ttf
Courier Newz.ttf      arialcb.ttf
# ln -s arialcb.ttf "Arialcb.ttf"
```

► Diesen Vorgang für jede einzelne erforderliche Font-Datei wiederholen.

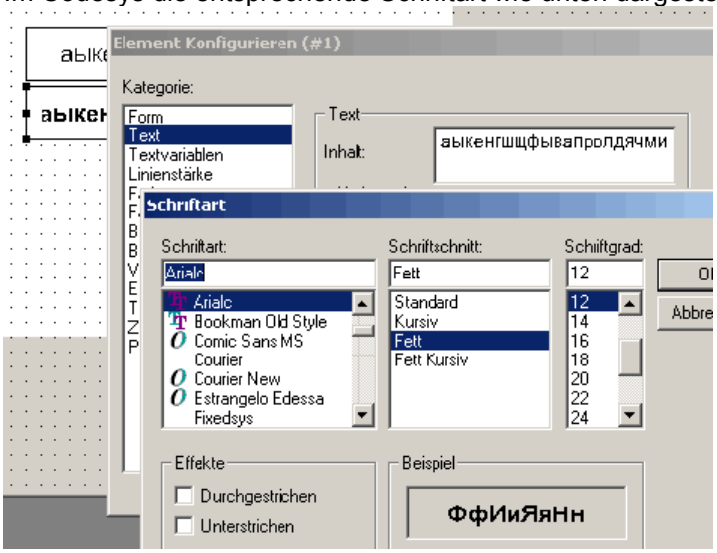
> Im FTP- Programm ergibt sich z.B. folgende Darstellung:



wobei hier oben das Verzeichnis und unten die Font-Datei stehen.

► Das PDM neu starten, damit die neuen Schriftarten zur Verfügung stehen.

► Im Codesys die entsprechende Schriftart wie unten dargestellt verwenden:



© ifm electronic gmbh

4.3.1 Symbol-Dateien

10073

ifm bietet Ihnen eine Reihe von Symbolen zum Einsatz in Ihrer Visualisierung an:
Auf der **ecomat*mobile***-DVD "Software, tools and documentation" oder – nach erfolgter CoDeSys-Installation – auf Ihrer Festplatte im Verzeichnis `...Targets\ifm\Library\Fonts\ifm_ISO` finden Sie die Dateien `ifm_iso_a.ttf` ... `ifm_iso_z.ttf`. Eine Übersicht und Erläuterungen zu den ISO-Dateien finden Sie in der Datei `ifm_iso- documentation.pdf` im selben Verzeichnis.
Zum Beispiel zu `ifm_iso_f.ttf`:

Annex F Engine

Key	Symbol No.	Symbol form/shape	Symbol description
!	F.01		Engine
"	F.02		Engine failure
#	F.03		Engine coolant temperature
$	F.04		Engine oil
%	F.05		Engine oil temperature
&	F.06		Engine oil level
'	F.07		Engine oil filter
(	F.08		Engine heating
)	F.09		Electronic diesel control
*	F.10		Diesel pre-heat
+	F.11		Choke
,	F.12		Hand throttle
-	F.13		Engine start
.	F.14		Engine shut-off
/	F.15		Engine coolant level
0	F.16		Engine coolant heating
1	F.17		Engine coolant fan
2	F.18		Engine inlet air filter
3	F.19		Engine inlet air pre-heat
4	F.20		Turbo
5	F.21		Engine emission filter
6	F.23		Turbocharger temperature
7	F.25		Engine status information
8	F.22		Engine emission system failure
9	F.24		Nox reduction agent
:	F.26		Engine emission system temperature

4.4 Programmiersystem einrichten

Inhalt:

Programmiersystem manuell einrichten 39
Programmiersystem über Templates einrichten 44
ifm-Demo-Programme 53

3968

4.4.1 Programmiersystem manuell einrichten

Inhalt:

Target einrichten 39
Steuerungskonfiguration aktivieren (z.B. CR1051) 41

3963

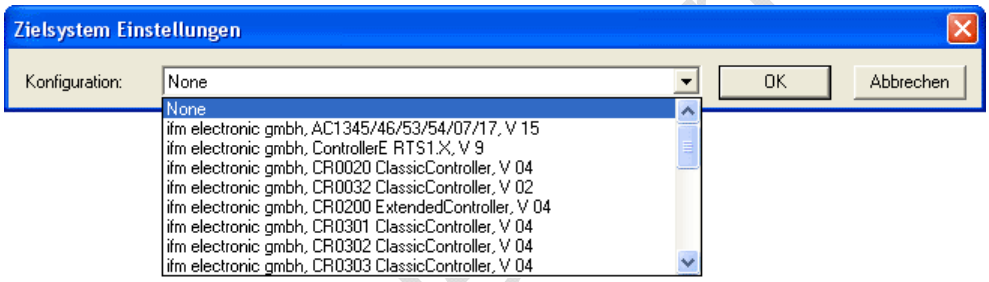
Target einrichten

7431

11379

Beim Erstellen eines neuen Projektes in CoDeSys muss die dem Gerät entsprechende Target-Datei geladen werden.

► Die gewünschte Target-Datei im Dialogfenster wählen (→ Screenshot).


Zielsystem Einstellungen
Konfiguration:
None
OK
Abbrechen
None
ifm electronic gmbh, AC1345/46/53/54/07/17, V 15
ifm electronic gmbh, ControllerE RTS1.X, V 9
ifm electronic gmbh, CR0020 ClassicController, V 04
ifm electronic gmbh, CR0032 ClassicController, V 02
ifm electronic gmbh, CR0200 ExtendedController, V 04
ifm electronic gmbh, CR0301 ClassicController, V 04
ifm electronic gmbh, CR0302 ClassicController, V 04
ifm electronic gmbh, CR0303 ClassicController, V 04

Grafik: Zielsystem Einstellungen (Beispiel)

> Die Target-Datei stellt für das Programmiersystem die Schnittstelle zur Hardware her.

> Gleichzeitig mit Wahl des Targets werden automatisch einige wichtige Bibliotheken und die Steuerungskonfiguration geladen.

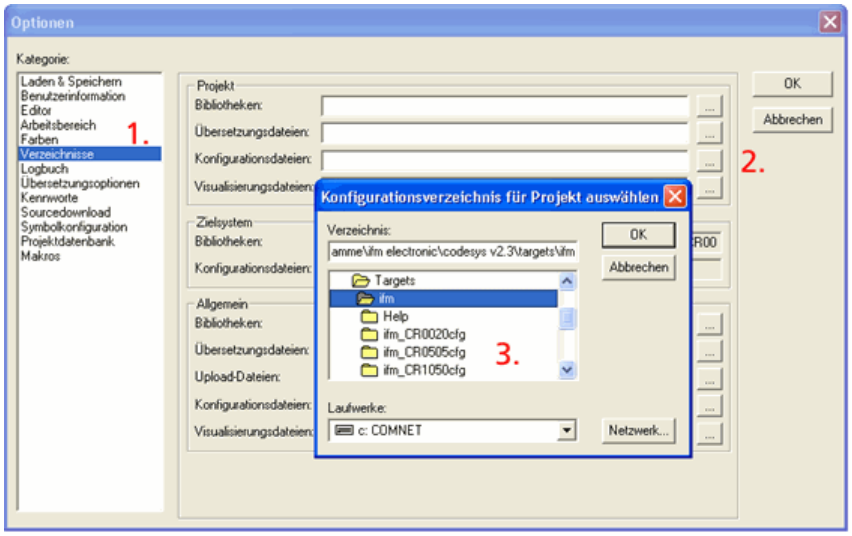
► Bei Bedarf geladene Bibliotheken wieder entfernen oder durch weitere Bibliotheken ergänzen.

► Immer die passende Geräte-Bibliothek `ifm_CRnnnn_Vxxyyzz.LIB` manuell ergänzen!

© ifm electronic gmbh

► Die entsprechende Steuerungskonfiguration wählen:

► Über das Menü [Projekt] > [Optionen...] das Fenster öffnen und den Punkt [Verzeichnisse...] wählen (→ (1.) im Bild).



Grafik: Konfigurationsverzeichnis für Projekt

► In der Verzeichnisliste neben der Eingabezeile [Projekt/Konfigurationsdateien:] auf die Schaltfläche […] klicken (→ (2.) im Bild).

> Explorer-Fenster erscheint: [Konfigurationsverzeichnis für Projekt auswählen].

► Das Quell-Verzeichnis mit dem passenden Target auswählen (→ (3.) im Bild)
z.B. `...\CoDeSys v2.3\Targets\ifm\CR1051cfg`.

► Verzeichnis mit der gewünschten Versionsnummer der Konfigurationsdatei auswählen, z.B. V030101.

► Mit 2x [OK] Vorgang abschließen.

HINWEIS

Es müssen immer die zum gewählten Target passenden Software-Stände zum Einsatz kommen:
- der Steuerungskonfiguration (`ifm_CRnnnn_Vxxyyyzz.CFG`),
- der Gerätebibliothek (`ifm_CRnnnn_Vxxyyzz.LIB`) und
- der weiteren Dateien (→ Kapitel *Übersicht der verwendeten Dateien und Bibliotheken*, → Seite 322).

CRnnnn	Geräte-Artikelnummer
Vxx: 00...99	Versionsnummer
yy: 00...99	Release-Nummer
zz: 00...99	Patch-Nummer

Dabei müssen der Basisdateiname (z.B. "`CR1051`") und die Software-Versionsnummer "xx" (z.B. "`04`") überall den gleichen Wert haben! Andernfalls geht das Gerät in den STOP-Zustand.

Die Werte für "yy" (Release-Nummer) und "zz" (Patch-Nummer) müssen **nicht** übereinstimmen.

Folgende Dateien müssen ebenfalls geladen sein:
- die zum Projekt erforderlichen internen Bibliotheken (in IEC 1131 erstellt),
- die Konfigurationsdateien (`*.CFG`)
- und die Target-Dateien (`*.TRG`).

Es kann vorkommen, dass das Zielsystem mit Ihrer aktuell installierten Version von CoDeSys nicht oder nur teilweise programmiert werden kann. Im diesem Fall wenden Sie sich bitte an den technischen Support der **ifm electronic gmbh**.

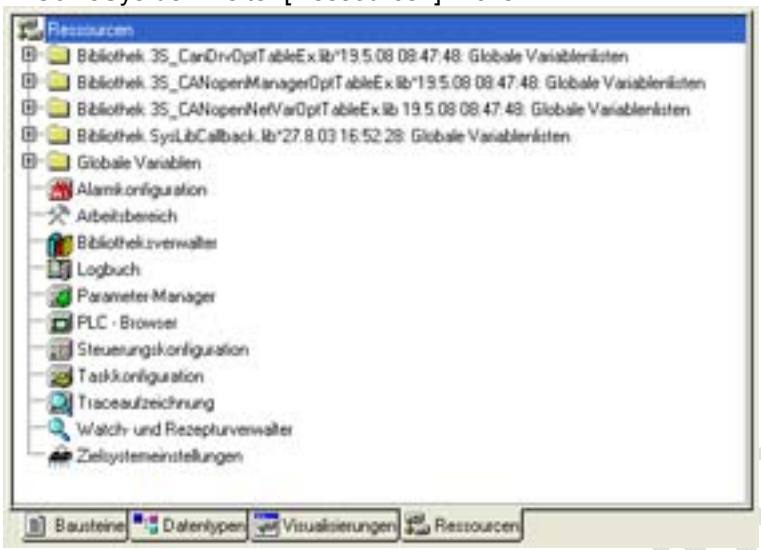
www.ifm electronic gmbh

Steuerungskonfiguration aktivieren (z.B. CR1051)

3045

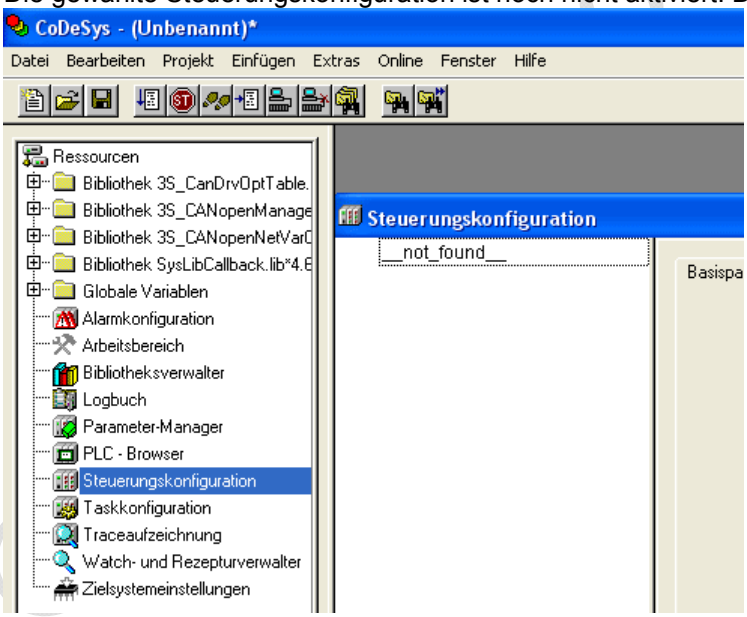
Den Punkt [Steuerungskonfiguration] erreicht man über den Reiter [Ressourcen]. Über einen Doppelklick auf den Punkt [Steuerungskonfiguration] öffnet sich das entsprechende Fenster.

► In CoDeSys den Reiter [Ressourcen] klicken:



► In der linken Spalte Doppelklick auf [Steuerungskonfiguration]

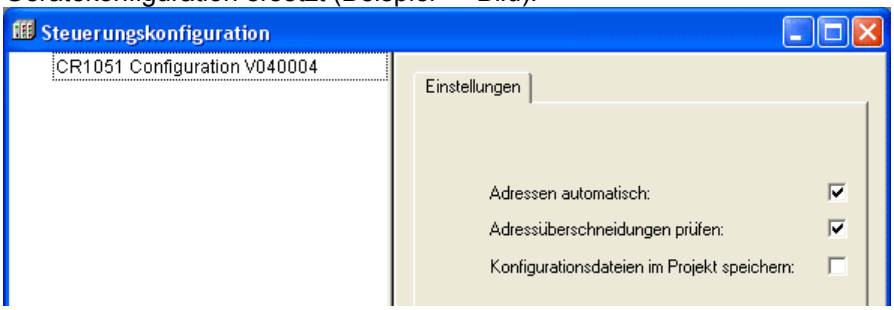
> Die gewählte Steuerungskonfiguration ist noch nicht aktiviert. Daher Anzeige: "__not_found__"



► Oben im Menü klicken auf [Extras] > [Standardkonfiguration]

► Sicherheitsabfrage mit Klick auf [Ja] bestätigen

> Der Eintrag "__not_found__" wird durch den Namen und die Version der zuvor gewählten Gerätekonfiguration ersetzt (Beispiel → Bild):



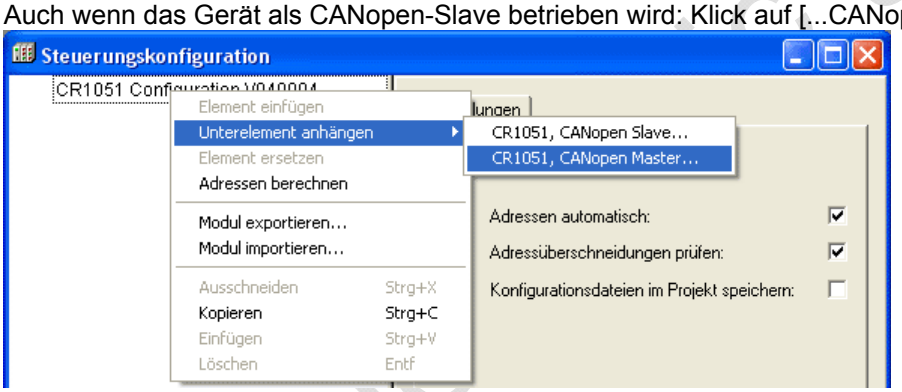
Durch die Konfiguration erhält der Anwender in der Programmumgebung Folgendes verfügbar:

- alle wichtigen System- und Fehlermerker
  Je nach Anwendung und Applikations-Programm müssen diese Merker bearbeitet und ausgewertet werden. Der Zugriff erfolgt über deren symbolischen Namen.

► Rechtsklick auf den Namen der Steuerungskonfiguration.

► Klick auf [Unterelement anhängen].

► Auch wenn das Gerät als CANopen-Slave betrieben wird: Klick auf [...CANopen Master...]:

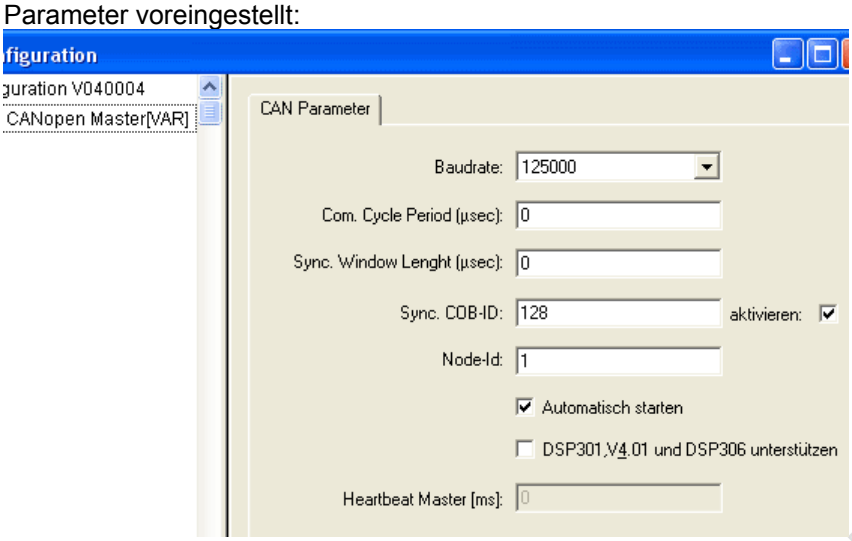


Wenn das Gerät als Slave betrieben wird, wäre die Auswahl [...CANopen Slave...] ebenfalls möglich.
→ Kapitel *CANopen-Slave*, → Seite 126

Bei der insgesamt einfacheren Konfiguration als Master können auch alle CAN-Layer2- und Netzwerkvariablen-Funktionen genutzt werden.

© ifm electronic gmbh

> Die CAN-Parameter der Steuerungskonfiguration erscheinen. Hier sind bereits einige CAN-Parameter voreingestellt:



► Wenn das Gerät via Netzwerkvariablen oder CAN_RX / CAN_TX auf CAN-Layer 2 oder als Slave betrieben wird:
Prüfen, ob für das Gerät die richtige Baudrate eingestellt ist (Baudrate muss für alle Teilnehmer identisch sein).

► Wenn das Gerät als CANopen-Master betrieben wird:
Alle Parameter-Einstellungen prüfen.

► Das Fenster [Steuerungskonfiguration] schließen.

► Mit Menü [Datei] > [speichern unter...] dem Projekt einen sinnvollen Namen geben und das Projekt im gewünschten Verzeichnis speichern.

© ifm electronic gmbh

4.4.2 Programmiersystem über Templates einrichten

Inhalt:

Über die ifm-Templates 46
Projekt mit weiteren Funktionen ergänzen 50

3977

ifm bietet vorgefertigte Templates (Programm-Vorlagen), womit Sie das Programmiersystem schnell, einfach und vollständig einrichten können.

Beim Installieren der **ecomatmobile**-DVD "Software, tools and documentation" wurden auch Projekte mit Vorlagen auf Ihrem Computer im Programmverzeichnis abgelegt:
…`\ifm electronic\CoDeSys V…\Projects\Template_CDV`…

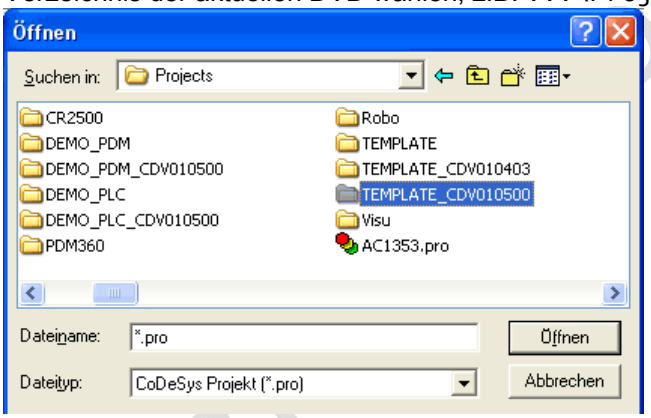
► Die gewünschte dort gespeicherte Vorlage in CoDeSys öffnen mit:
[Datei] > [Neu aus Vorlage…]

> CoDeSys legt ein neues Projekt an, dem der prinzipielle Programmaufbau entnommen werden kann. Es wird dringend empfohlen, dem gezeigten Schema zu folgen.
→ Kapitel *Programmiersystem über Templates einrichten*, → Seite 44

Wie richten Sie das Programmiersystem schnell und einfach ein? (z.B. CR2500)

► Im CoDeSys-Menü wählen: [Datei] > [Neu aus Vorlage...].

► Verzeichnis der aktuellen DVD wählen, z.B. `...\Projects\TEMPLATE_CDV010500`:



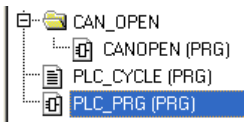
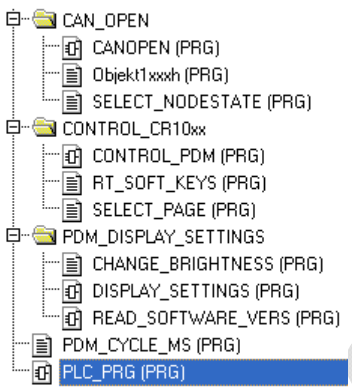
► Artikelnummer des Geräts in der Liste suchen, z.B. CR2500 als CANopen-Master:

► Achten Sie auch auf die richtige Programm-Version!

► Wie ist das CAN-Netzwerk organisiert?
Soll auf Layer2-Basis gearbeitet werden oder gibt es (mit CANopen) einen Master mit mehreren Slaves?

► Wahl mit [Öffnen] bestätigen.

> Neues CoDeSys-Projekt wird angelegt mit zunächst folgender Ordnerstruktur (links):

Beispiel für CR2500 als CANopen-Master:	Anderes Beispiel für CR1051 als CANopen-Slave:
	

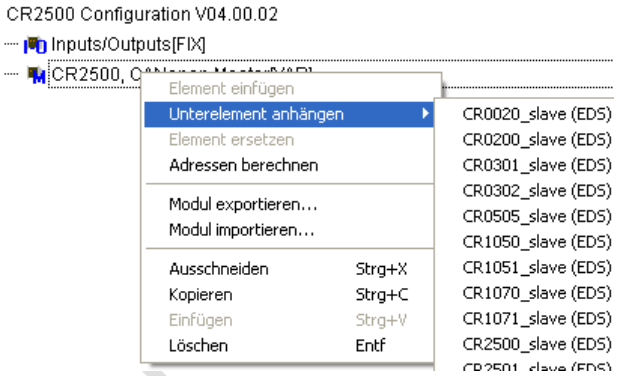
(Über die Ordnerstrukturen in Templates → Kapitel *Über die ifm-Templates*, → Seite 46).

► Das neue Projekt speichern mit [Datei] > [Speichern unter...], dabei geeignetes Verzeichnis und Projektnamen festlegen.

► Das CAN-Netzwerk im Projekt konfigurieren:
Im CoDeSys-Projekt über dem Tabulator [Ressourcen] das Element [Steuerungskonfiguration] doppelklicken.

► Mit **rechter** Maustaste in den Eintrag [CR2500, CANopen Master] klicken.

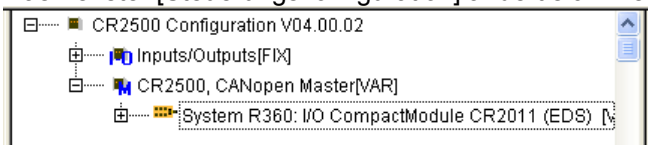
► Im Kontext-Menü [Unterelement anhängen] klicken:



> Im ergänzten Kontextmenü erscheint eine Liste aller verfügbaren EDS-Dateien.

► Gewünschtes Element wählen, z.B. "System R360: I/O CompactModule CR2011 (EDS)".
Die EDS-Dateien liegen im Verzeichnis `C:\…\CoDeSys V…\Library\PLCConf\`.

> Das Fenster [Steuerungskonfiguration] ändert sich wie folgt:



► Für den eingetragenen Slave den Erfordernissen entsprechend die CAN-Parameter, das PDO-Mapping und die SDOs einstellen.
[alle SDOs erzeugen] besser abwählen.

► Mit weiteren Slaves sinngemäß wie vorstehend verfahren.

► Projekt speichern!

Damit ist das Netzwerk Ihres Projekts hinreichend beschrieben. Sie wollen dieses Projekt mit weiteren Elementen und Funktionen ergänzen?
→ Kapitel *Projekt mit weiteren Funktionen ergänzen*, → Seite 50

Über die ifm-Templates

Inhalt:

Ordner-Struktur, allgemein 46
Programme und Funktionen in den Ordnern der Templates 47
Struktur der Visualisierungen in den Templates 49

3981

In der Regel werden für jedes Gerät folgende Templates angeboten:

- `ifm_template_CRnnnnLayer2_Vxxyyzz.pro`
  für den Betrieb des Geräts mit CAN Layer 2
- `ifm_template_CRnnnnMaster_Vxxyyzz.pro`
  für den Betrieb des Geräts als CANopen-Master
- `ifm_template_CRnnnnSlave_Vxxyyzz.pro`
  für den Betrieb des Geräts als CANopen-Slave

Die hier beschriebenen Templates gelten für:
- CoDeSys ab Version 2.3.9.6
- auf der **ecomat*mobile***-DVD "Software, tools and documentation" ab Version 010500

Die Templates enthalten alle die gleichen Strukturen.

Mit dieser Auswahl der Programm-Vorlage für den CAN-Betrieb ist bereits eine wichtige Grundlage für ein funktionsfähiges Programm geschaffen.

Ordner-Struktur, allgemein

3978

Die Bausteine sind sortiert in die folgenden Ordner:

Ordner	Beschreibung
CAN_OPEN	für Controller und PDM, CAN-Betrieb als Master oder Slave: Enthält die Bausteine für CANopen.
I_O_CONFIGURATION	für Controller, CAN-Betrieb mit Layer 2 oder als Master oder als Slave: Bausteine zum Parametrieren der Betriebsarten der Ein- und Ausgänge.
PDM_COM_LAYER2	für Controller, CAN-Betrieb als Layer 2 oder Slave: Bausteine zur Basiskomunikation über Layer2 zwischen PLC und PDM.
CONTROL_CR10nn	für PDM, CAN-Betrieb mit Layer 2 oder als Master oder als Slave: Enthält Bausteine zur Bild- und Tastensteuerung im laufenden Betrieb.
PDM_DISPLAY_SETTINGS	für PDM, CAN-Betrieb mit Layer 2 oder als Master oder als Slave: Enthält Bausteine zum Einstellen des Monitors.

© ifm electronic gmbh

Programme und Funktionen in den Ordnern der Templates

3980

Die vorgenannten Ordner enthalten die folgenden Programme und Bausteine:

Bausteine im Ordner CAN_OPEN	**Beschreibung**
CANOPEN	für Controller und PDM, CAN-Betrieb als Master: Enthält folgende parametrierte Bausteine: - CAN1_MASTER_EMCY_HANDLER (→ *CANx_MASTER_EMCY_HANDLER*, → Seite 142), - CAN1_MASTER_STATUS (→ *CANx_MASTER_STATUS*, → Seite 147), - SELECT_NODESTATE (→ unten).
CANOPEN	für Controller und PDM, CAN-Betrieb als Slave: Enthält folgende parametrierte Bausteine: - CAN1_SLAVE_EMCY_HANDLER (→ *CANx_SLAVE_EMCY_HANDLER*, → Seite 155), - CAN1_SLAVE_STATUS (→ *CANx_SLAVE_STATUS*, → Seite 160), - SELECT_NODESTATE (→ unten).
Objekt1xxxh	für Controller und PDM, CAN-Betrieb als Slave: Enthält die Werte [STRING] zu folgenden Parametern: - ManufacturerDeviceName, z.B.: 'CR1051' - ManufacturerHardwareVersion, z.B.: 'HW_Ver 1.0' - ManufacturerSoftwareVersion, z.B.: 'SW_Ver 1.0'
SELECT_NODESTATE	für PDM, CAN-Betrieb als Master oder als Slave: Wandelt den Wert des Knoten-Status [BYTE] in den zugehörigen Text [STRING]: 4 ⇨ 'STOPPED' 5 ⇨ 'OPERATIONAL' 127 ⇨ 'PRE-OPERATIONAL'

Bausteine im Ordner I_O_CONFIGURATION	**Beschreibung**
CONF_IO_CRnnnn	für Controller, CAN-Betrieb mit Layer 2 oder als Master oder als Slave: Parametriert die Betriebsarten der Ein- und Ausgänge.

Bausteine im Ordner PDM_COM_LAYER2	**Beschreibung**
PLC_TO_PDM	für Controller, CAN-Betrieb mit Layer 2 oder als Slave: Organisiert die Kommunikation vom Controller zum PDM: - überwacht die Übertragungszeit, - überträgt Steuerdaten für Bildwechsel, LEDs, Eingabewerte usw.
TO_PDM	für Controller, CAN-Betrieb mit Layer 2 oder als Slave: Organisiert die Signale für LEDs und Tasten zwischen Controller und PDM. Enthält folgende parametrierte Bausteine: - PACK (→ 3S), - PLC_TO_PDM (→ oben), - UNPACK (→ 3S).

© ifm electronic gmbh

Bausteine im Ordner CONTROL_CR10nn	Beschreibung
CONTROL_PDM	für PDM, CAN-Betrieb mit Layer 2 oder als Master oder als Slave: Organisiert die Bildsteuerung im PDM. Enthält folgende parametrierte Bausteine: - PACK (→ 3S), - PDM_MAIN_MAPPER (→ *PDM_MAIN_MAPPER*, → Seite 255), - PDM_PAGECONTROL (→ *PDM_PAGECONTROL*, → Seite 259), - PDM_TO_PLC (→ unten), - SELECT_PAGE (→ unten).
PDM_TO_PLC	für PDM, CAN-Betrieb mit Layer 2: Organisiert die Kommunikation vom PDM zum Controller: - überwacht die Übertragungszeit, - überträgt Steuerdaten für Bildwechsel, LEDs, Eingabewerte usw. Enthält folgende parametrierte Bausteine: - CAN_1_TRANSMIT (→ *CAN_x_TRANSMIT*, → Seite 99), - CAN_1_RECEIVE (→ *CAN_x_RECEIVE*, → Seite 97).
RT_SOFT_KEYS	für PDM, CAN-Betrieb mit Layer 2 oder als Master oder als Slave: Liefert von den (virtuellen) Tasten-Signalen im PDM die steigenden Flanken. Es können beliebige Variablen (als virtuelle Tasten) auf die globalen Variablen SoftKeyGlobal gemappt werden, wenn z.B. ein Programmteil von einem CR1050 in ein CR1055 kopiert werden soll. Dort gibt es nur die Tasten F1...F3: ▶ Für die virtuellen Tasten F4...F6 Variablen erzeugen. Diese selbst erzeugten Variablen hier auf die globalen Softkeys mappen. Im Programm nur mit den globalen Softkeys arbeiten. Vorteil: Anpassungsarbeiten sind nur an **einer** Stelle erforderlich.
SELECT_PAGE	für PDM, CAN-Betrieb mit Layer 2 oder als Master oder als Slave: Organisiert die Wahl der Visualisierungen. Enthält folgende parametrierte Bausteine: - RT_SOFT_KEYS (→ oben).

Bausteine im Ordner PDM_DISPLAY_SETTINGS	Beschreibung
CHANGE_BRIGHTNESS	für PDM, CAN-Betrieb mit Layer 2 oder als Master oder als Slave: Organisiert Helligkeit / Kontrast des Monitors.
DISPLAY_SETTINGS	für PDM, CAN-Betrieb mit Layer 2 oder als Master oder als Slave: Stellt die Echtzeituhr, steuert Helligkeit / Kontrast des Monitors, zeigt die Software-Version. Enthält folgende parametrierte Bausteine: - CHANGE_BRIGHTNESS (→ oben), - CurTimeEx (→ 3S), - PDM_SET_RTC (→ *PDM_SET_RTC*, → Seite 188), - READ_SOFTWARE_VERS (→ unten), - TP (→ 3S).
READ_SOFTWARE_VERS	für PDM, CAN-Betrieb mit Layer 2 oder als Master oder als Slave: Zeigt die Software-Version. Enthält folgende parametrierte Bausteine: - DEVICE_KERNEL_VERSION1 (→ *DEVICE_KERNEL_VERSION1*), - DEVICE_RUNTIME_VERSION (→ *DEVICE_RUNTIME_VERSION*, → Seite 192), - LEFT (→ 3S).

© ifm electronic gmbh

Bausteine im Wurzel-Verzeichnis	Beschreibung
PLC_CYCLE	für Controller, CAN-Betrieb mit Layer 2 oder als Master oder als Slave: Ermittelt die Zykluszeit der SPS im Gerät.
PDM_CYCLE_MS	für PDM, CAN-Betrieb mit Layer 2 oder als Master oder als Slave: Ermittelt die Zykluszeit der SPS im Gerät.
PLC_PRG	für Controller und PDM, CAN-Betrieb mit Layer 2 oder als Master oder als Slave: Hauptprogramm; hier werden die weiteren Programm-Elemente eingebunden.

Struktur der Visualisierungen in den Templates

3979

Für folgende Geräte verfügbar:
- BasicDisplay: CR0451
- PDM: CR10nn

Die Visualisierungen sind wie folgt in Ordnern strukturiert:

Ordner	Bild-Nr.	Beschreibung Inhalt
START_PAGE	P00001	Einstellung / Anzeige von... - Node-ID - CAN-Baudrate - Status - GuardErrorNode - SPS-Zykluszeit
__MAIN_MENUES	P00010	Menübild: - Display-Setup
____MAIN_MENUE_1		
______DISPLAY_SETUP		
________1_DISPLAY_SETUP1	P65000	Menübild: - Software-Version - Helligkeit / Kontrast - Echtzeituhr anzeigen / setzen
__________1_SOFTWARE_VERSION	P65010	Anzeige der Software-Version
__________2_BRIGHTNESS	P65020	Einstellen von Helligkeit / Kontrast
__________3_SET_RTC	P65030	Echtzeituhr anzeigen / setzen

In den Templates haben wir die Bildnummern in 10er-Schritten organisiert. So können Sie mit Hilfe eines Bildnummer-Offsets in verschiedene Sprachversionen der Visualisierungen schalten.

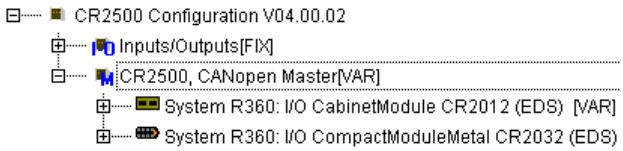
© ifm electronic gmbh

Projekt mit weiteren Funktionen ergänzen

3987

Sie haben ein Projekt mittels eines ifm-Templates angelegt und das CAN-Netzwerk definiert. Nun wollen Sie diesem Projekt weitere Funktionen hinzufügen.

Für das Beispiel nehmen wir einen CabinetController CR2500 als CANopen-Master an, an den ein I/O-CabinetModul CR2011 und ein I/O-Compact-Modul CR2032 als Slaves angeschlossen sind:

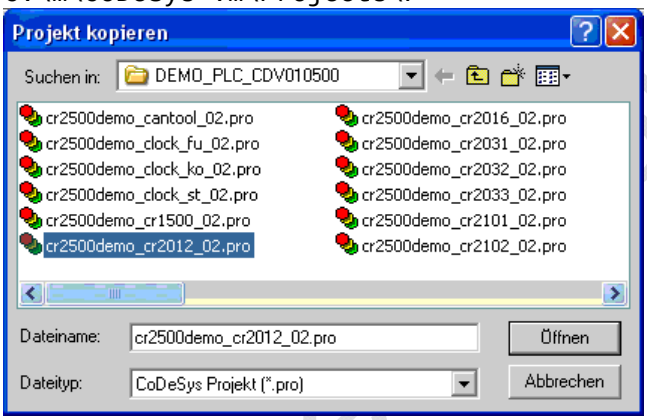

CR2500 Configuration V04.00.02
Inputs/Outputs[FIX]
CR2500, CANopen Master[VAR]
System R360: I/O CabinetModule CR2012 (EDS) [VAR]
System R360: I/O CompactModuleMetal CR2032 (EDS)

Beispiel: Steuerungskonfiguration

Am CR2012 sei ein Joystick angeschlossen, der am CR2032 einen PWM-Ausgang ansteuern soll. Wie geht das schnell und einfach?

► CoDeSys-Projekt speichern!

► In CoDeSys mit [Projekt] > [kopieren...] das Projekt öffnen, das die gewünschte Funktion enthält: z.B. `CR2500Demo_CR2012_02.pro` aus dem Verzeichnis `DEMO_PLC_CDV…` unter `C:\…\CoDeSys V…\Projects\`:

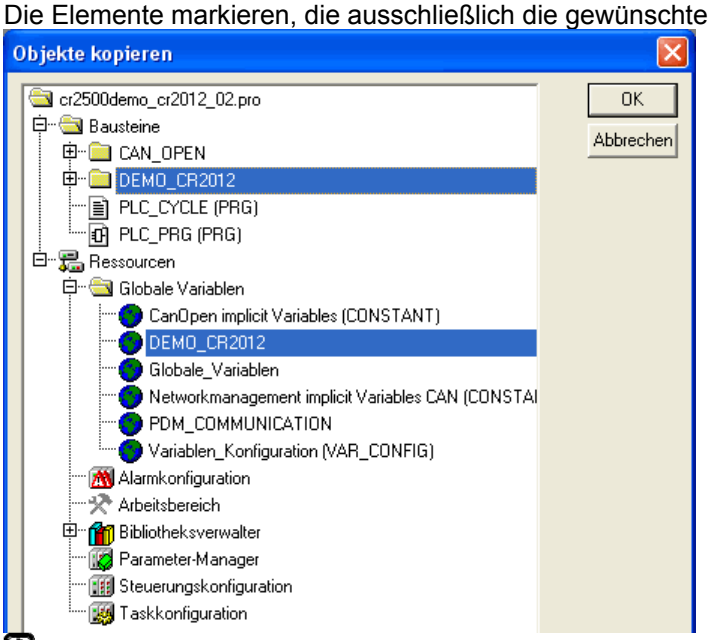


► Wahl mit [Öffnen] bestätigen.

► Die Meldung "Fehler beim Laden der Steuerungskonfiguration" kann ignoriert werden.

> Fenster [Objekte kopieren] erscheint:

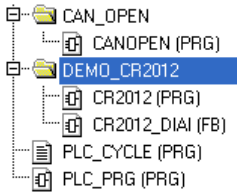
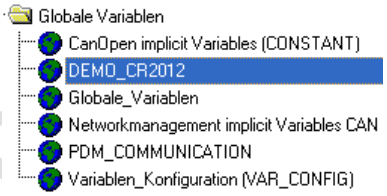
► Die Elemente markieren, die ausschließlich die gewünschte Funktion enthalten, hier z.B.:



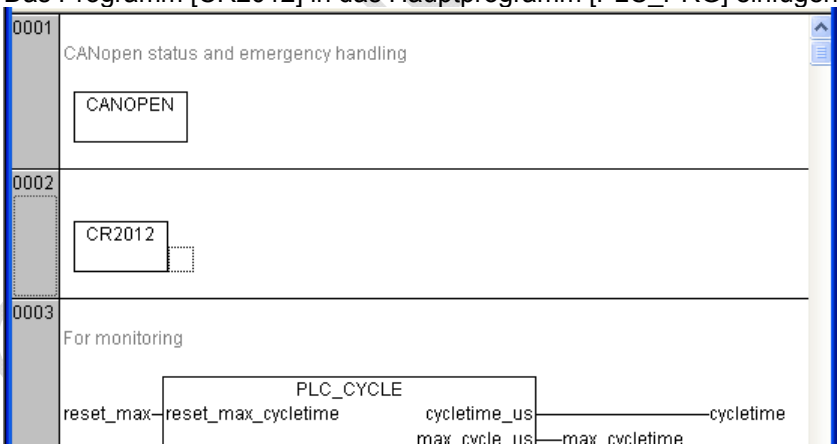
In anderen Fällen können auch Bibliotheken und / oder Visualisierungen erforderlich sein.

► Wahl mit [OK] bestätigen.

> In unserem Beispiel-Projekt sind die im Demo-Projekt gewählten Elemente hinzugekommen:

Bausteine:	Ressourcen:
	

► Das Programm [CR2012] in das Hauptprogramm [PLC_PRG] einfügen, z.B.:



► In den Kommentaren der Bausteine und Globalen Variablen stehen meist Hinweise, wie bei Bedarf einzelne Elemente daraus konfiguriert, eingeschlossen oder ausgeschlossen werden müssen. Diesen Hinweisen Folge leisten.

► Ein- und Ausgangsvariable sowie CAN-Parameter und ggf. Visualisierungen den eigenen Bedingungen anpassen.

► [Projekt] > [speichern] und
[Projekt] > [Alles übersetzen].

- ► Nach eventuell erforderlichen Korrekturen und Ergänzen von fehlenden Bibliotheken (→ Fehlermeldungen nach dem Übersetzen) das Projekt nochmals speichern.
- ► Nach diesem Prinzip schrittweise (!) mit weiteren Funktionen aus anderen Projekten ergänzen und jeweils die Ergebnisse prüfen.
- ► [Projekt] > [speichern] und
  [Projekt] > [Alles übersetzen].

© ifm electronic gmbh

4.4.3 ifm-Demo-Programme

Inhalt:

Demo-Programme für Controller 53
Demo-Programme für PDM und BasicDisplay 55

3982

Im Verzeichnis `DEMO_PLC_CDV…` (für Controller) oder `DEMO_PDM_CDV…` (für PDMs) unter `C:\…\CoDeSys V…\Projects\` erklären wir bestimmte Funktionen in getesteten Demo-Programmen. Bei Bedarf können diese Funktionen in eigene Projekte übernommen werden. Die Strukturen und Variablen der **ifm**-Demos passen zu denen in den **ifm**-Templates.

In jedem Demo-Programm wird nur genau **ein** Thema gezeigt. Auch für Controller werden dazu einige Visualisierungen gezeigt, die auf dem PC-Monitor die getestete Funktion anschaulich machen sollen.

Kommentare in den Bausteinen und in den Variablenlisten helfen beim Anpassen der Demos an Ihr Projekt.

Wenn nicht anders angegeben, gelten die Demo-Programme jeweils für alle Controller oder für alle PDMs.

Die hier beschriebenen Demo-Programme gelten für:
- CoDeSys ab Version 2.3.9.6
- auf der **ecomat*mobile***-DVD "Software, tools and documentation" ab Version 010500

Demo-Programme für Controller

3995

Demo-Programm	Funktion
`CR2500Demo_CanTool_xx.pro`	getrennt für PDM360, PDM360compact, PDM360smart und Controller: Enthält Funktionen zum Einstellen und Analysieren der CAN-Schnittstelle.

`CR2500Demo_ClockFu_xx.pro`
`CR2500Demo_ClockKo_xx.pro`
`CR2500Demo_ClockSt_xx.pro`
	Taktgenerator für Controller als Funktion eines Wertes an einem Analog-Eingang:
	Fu = in Funktionsplan
	Ko = in Kontaktplan
	St = in Strukturiertem Text

`CR2500Demo_CR1500_xx.pro`	Anschluss eines Tastatur-Moduls CR1500 als Slave eines Controllers (CANopen-Master).
`CR2500Demo_CR2012_xx.pro`	I/O-Cabinet-Modul CR2012 als Slave eines Controllers (CANopen-Master), Anschluss eines Joysticks mit Richtungsschalter und Referenz-Mittelspannung.
`CR2500Demo_CR2016_xx.pro`	I/O-Cabinet-Modul CR2016 als Slave eines Controllers (CANopen-Master), 4x Frequenz-Eingang, 4x Digital-Eingang Highside, 4x Digital-Eingang Lowside, 4x Analog-Eingang ratiometrisch, 4x PWM1000-Ausgang und 12x Digitalausgang.
`CR2500Demo_CR2031_xx.pro`	I/O-Compact-Modul CR2031 als Slave eines Controllers (CANopen-Master), Strommessung an den PWM-Ausgängen.

© ifm electronic gmbh

Demo-Programm	Funktion
`CR2500Demo_CR2032_xx.pro`	I/O-Compact-Modul CR2032 als Slave eines Controllers (CANopen-Master), 4x Digital-Eingang, 4x Digital-Eingang analog ausgewertet, 4x Digital-Ausgang, 4x PWM-Ausgang.
`CR2500Demo_CR2033_xx.pro`	I/O-Compact-Modul CR2033 als Slave eines Controllers (CANopen-Master), 4x Digital-Eingang, 4x Digital-Eingang analog ausgewertet, 4x Digital-Ausgang.
`CR2500Demo_CR2101_xx.pro`	Neigungssensor CR2101 als Slave eines Controllers (CANopen-Master).
`CR2500Demo_CR2102_xx.pro`	Neigungssensor CR2102 als Slave eines Controllers (CANopen-Master).
`CR2500Demo_CR2511_xx.pro`	I/O-Smart-Modul CR2511 als Slave eines Controllers (CANopen-Master), 8x PWM-Ausgang stromgeregelt.
`CR2500Demo_CR2512_xx.pro`	I/O-Smart-Modul CR2512 als Slave eines Controllers (CANopen-Master), 8x PWM-Ausgang. Anzeige des aktuellen Stroms für jedes Kanalpaar.
`CR2500Demo_CR2513_xx.pro`	I/O-Smart-Modul CR2513 als Slave eines Controllers (CANopen-Master), 4x Digital-Eingang, 4x Digital-Ausgang, 4x Analogeingang 0...10 V.
`CR2500Demo_Interrupt_xx.pro`	Beispiel mit *SET_INTERRUPT_XMS*.
`CR2500Demo_Operating_hours_xx.pro`	Beispiel für einen Betriebsstundenzähler mit Schnittstelle zu einem PDM.
`CR2500Demo_PWM_xx.pro`	Wandelt einen Potentiometer-Wert an einem Eingang in einen normierten PWM-Wert an einem Ausgang mit folgenden Bausteinen: - *INPUT_VOLTAGE*, - *NORM*, - *PWM100*.
`CR2500Demo_RS232_xx.pro`	Beispiel für den Empfang von Daten auf der seriellen Schnittstelle mit Hilfe des Windows-Hyperterminal.

`StartersetDemo.pro`
`StartersetDemo2.pro`
`StartersetDemo2_fertig.pro`
	Verschiedene Übungen zum E-Learning mit dem Starterset EC2074.

_xx = Angabe der Demo-Version

© ifm electronic gmbh

Demo-Programme für PDM und BasicDisplay

3996

Demo-Programm	Funktion

`CR1051Demo_CanTool_xx.pro`
`CR1053Demo_CanTool_xx.pro`
`CR1071Demo_CanTool_xx.pro`
	getrennt für PDM360, PDM360compact, PDM360smart und Controller:
	
	Enthält Funktionen zum Einstellen und Analysieren der CAN-Schnittstelle.

`CR1051Demo_Input_Character_xx.pro`	Ermöglicht beliebige Zeicheneingabe in eine Zeichenkette: - Großbuchstaben, - Kleinbuchstaben, - Sonderzeichen, - Ziffern. Auswahl der Zeichen mit dem Drehgeber. Beispiel ist auch z.B. für eine Passworteingabe geeignet. Bild P01000: Auswahl und Übernahme von Zeichen
`CR1051Demo_Input_Lib_xx.pro`	Demo von *INPUT_INT* (→ Seite 272) aus der Bibliothek `ifm_pdm_input_Vxxyyzz` (mögliche Alternative zum 3S-Standard). Werte wählen und einstellen mittels Drehgeber. Bild P10000: 6 Werte INT Bild P10010: 2 Werte INT Bild P10020: 1 Wert REAL
`CR1051Demo_Linear_logging_on_flash_intern_xx.pro`	Schreibt einen CSV-Datensatz mit dem Inhalt einer CAN-Nachricht in den internen Flash-Speicher (`/home/project/daten.csv`), wenn [F3] gedrückt wird oder eine CAN- Nachricht auf dem ID 100 empfangen wurde. Wenn der definierte Speicherbereich gefüllt ist, wird die Aufzeichnung der Daten beendet. Verwendete Bausteine: - *WRITE_CSV_8BYTE* (→ Seite 207), - *SYNC* (→ Seite 200). Bild P35010: Anzeige Datei-Informationen Bild P35020: Anzeige aktueller Datensatz Bild P35030: Anzeige Liste von 10 Datensätzen
`CR1051Demo_O2M_1Cam_xx.pro`	Anschluss von 1 Kamera O2M100 am Monitor mit *CAM_O2M*. Umschalten zwischen Teil- und Vollbild. Bild 39000: Auswahlmenü Bild 39010: Kamerabild + Textbox Bild 39020: Kamerabild als Vollbild Bild 39030: nur Visualisierung
`CR1051Demo_O2M_2Cam_xx.pro`	Anschluss von 2 Kameras O2M100 am Monitor mit *CAM_O2M*. Umschalten zwischen den Kameras und zwischen Teil- und Vollbild. Bild 39000: Auswahlmenü Bild 39010: Kamerabild + Textbox Bild 39020: Kamerabild als Vollbild Bild 39030: nur Visualisierung
`CR1051Demo_Powerdown_Retain_bin_xx.pro`	Beispiel mit *PDM_POWER_DOWN* aus der Bibliothek `ifm_CR1051_Vxxyyzz.Lib`, um Retain-Variable in die Datei `Retain.bin` zu speichern. Simulation des ShutDown mit [F3].
`CR1051Demo_Powerdown_Retain_bin2_xx.pro`	Beispiel mit *PDM_POWER_DOWN* aus der Bibliothek `ifm_CR1051_Vxxyyzz.Lib`, um Retain-Variable in die Datei `Retain.bin` zu speichern. Simulation des ShutDown mit [F3].
`CR1051Demo_Powerdown_Retain_cust_xx.pro`	Beispiel mit *PDM_POWER_DOWN* und *PDM_READ_RETAIN* aus der Bibliothek `ifm_CR1051_Vxxyyzz.Lib`, um Retain-Variable in die Datei `/home/project/myretain.bin` zu speichern. Simulation des ShutDown mit [F3].

© ifm electronic gmbh

Demo-Programm	Funktion
`CR1051Demo_Read_Textline_xx.pro`	Das Beispiel-Programm liest jeweils 7 Textzeilen aus dem PDM-Dateisystem mit Hilfe von *READ_TEXTLINE* (→ Seite 203). Bild P01000: Anzeige gelesener Text
`CR1051Demo_Real_in_xx.pro`	Einfaches Beispiel für die Eingabe eines REAL-Werts in das PDM. Bild P01000: Eingabe und Anzeige des REAL-Werts
`CR1051Demo_Ringlogging_on_flash_intern_xx.pro`	Schreibt einen CSV-Datensatz in den internen Flash-Speicher, wenn [F3] gedrückt wird oder eine CAN-Nachricht auf dem ID 100 empfangen wurde. Die Dateinamen sind frei definierbar. Wenn der definierte Speicherbereich gefüllt ist, beginnt die Aufzeichnung der Daten von vorn. Verwendete Bausteine: - *WRITE_CSV_8BYTE* (→ Seite 207), - *SYNC* (→ Seite 200). Bild P35010: Anzeige Datei-Informationen Bild P35020: Anzeige aktueller Datensatz Bild P35030: Anzeige Liste von 8 Datensätzen
`CR1051Demo_Ringlogging_on_flash_pcmcia_xx.pro`	Schreibt einen CSV-Datensatz auf die PCMCIA-Karte, wenn [F3] gedrückt wird oder eine CAN-Nachricht auf dem ID 100 empfangen wurde. Die Dateinamen sind frei definierbar. Wenn der definierte Speicherbereich gefüllt ist, beginnt die Aufzeichnung der Daten von vorn. Verwendete Bausteine: - *WRITE_CSV_8BYTE* (→ Seite 207), - *OPEN_PCMCIA* (→ Seite 198), - *SYNC* (→ Seite 200). Bild P35010: Anzeige Datei-Informationen Bild P35020: Anzeige aktueller Datensatz Bild P35030: Anzeige Liste von 8 Datensätzen
`CR1051Demo_RW-Parameter_xx.pro`	In einer Liste können Parameter gewählt und geändert werden. Beispiel mit folgenden Bausteinen: - *READ_PARAMETER_WORD* (→ Seite 210), - *WRITE_PARAMETER_WORD* (→ Seite 212). Bild P35010: Liste von 20 Parametern

_xx = Angabe der Demo-Version

© ifm electronic gmbh

4.5 Erste Schritte

Inhalt:

Fehlende Bibliotheken einfügen 57
Visualisierung erstellen 59
PLC-Programm erstellen 61
Projekt im Gerät speichern 62

3044

► Geräteparameter einstellen (→ *Geräteparameter einstellen (Setup)*, → Seite 11).

► Programmiersystem einrichten (→ *Programmiersystem einrichten*, → Seite 39).

4.5.1 Fehlende Bibliotheken einfügen

3098

Die Gerätedaten sind dem CoDeSys-Projekt bekannt, die Steuerungskonfiguration ist aktiviert. Einige Bibliotheken werden bereits automatisch geladen. Abhängig von der Applikation müssen Sie dem Projekt noch einige Bibliotheken hinzufügen. Die Beschreibung dazu folgt hier.

► In CoDeSys den Reiter [Ressourcen] klicken:

► In der linken Spalte Doppelklick auf [Bibliotheksverwalter]

► Mit Taste [Einfg] oder Menü [Einfügen] > [weitere Bibliothek ...] die Bibliotheks-Übersicht dieses Geräts anfordern.

> Das Fenster [Öffnen] erscheint mit der Bibliotheks-Übersicht.

© ifm electronic gmbh

Die hier gezeigten Bibliotheken haben folgende Funktionen:

Bibliothek	Bedeutung
`ifm_CRnnnn_CAN1openMaster_Vxxyyzz`	CANopen-Master für Schnittstelle CAN1
`ifm_CRnnnn_CAN1openSlave_Vxxyyzz`	CANopen-Slave für Schnittstelle CAN1
`ifm_CRnnnn_CAN2openMaster_Vxxyyzz`	CANopen-Master für Schnittstelle CAN2
`ifm_CRnnnn_CAN2openSlave_Vxxyyzz`	CANopen-Slave für Schnittstelle CAN2
`ifm_CRnnnn_Vxxyyzz`	Geräte-Bibliothek

HINWEIS

Es müssen immer die zum gewählten Target passenden Software-Stände zum Einsatz kommen:
- der Steuerungskonfiguration (`ifm_CRnnnn_Vxxyyyzz.CFG`),
- der Gerätebibliothek (`ifm_CRnnnn_Vxxyyzz.LIB`) und
- der weiteren Dateien (→ Kapitel *Übersicht der verwendeten Dateien und Bibliotheken*, → Seite 322).

CRnnnn	Geräte-Artikelnummer
Vxx: 00...99	Versionsnummer
yy: 00...99	Release-Nummer
zz: 00...99	Patch-Nummer

Dabei müssen der Basisdateiname (z.B. "`CR1051`") und die Software-Versionsnummer "xx" (z.B. "`04`") überall den gleichen Wert haben! Andernfalls geht das Gerät in den STOP-Zustand.

Die Werte für "yy" (Release-Nummer) und "zz" (Patch-Nummer) müssen **nicht** übereinstimmen.

Folgende Dateien müssen ebenfalls geladen sein:
- die zum Projekt erforderlichen internen Bibliotheken (in IEC 1131 erstellt),
- die Konfigurationsdateien (`*.CFG`)
- und die Target-Dateien (`*.TRG`).

Es kann vorkommen, dass das Zielsystem mit Ihrer aktuell installierten Version von CoDeSys nicht oder nur teilweise programmiert werden kann. Im diesem Fall wenden Sie sich bitte an den technischen Support der **ifm electronic gmbh**.

► Wenn noch nicht im Projekt integriert, nacheinander die folgenden Bibliotheken einfügen:
 - Standard-Bibliothek `Standard.Lib` aus `C:\...\CoDeSys\Library\`
 - Geräte-Bibliothek `CRnnnn_Vxxyyzz.Lib` aus `C:\...\CoDeSys\Targets\ifm\Library\ifm_CRnnnn\`

► Das Projekt mit [Strg]+[s] sichern.

> Das Projekt ist nun vorbereitet für das PLC-Programm der Applikation.

© ifm electronic gmbh

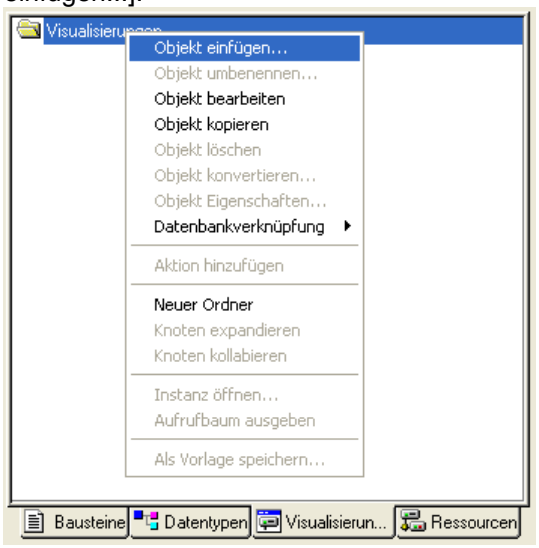
4.5.2 Visualisierung erstellen

3100

Für dieses Beispiel erstellen wir zuerst die Visualisierung, erst anschließend das PLC-Programm dazu.

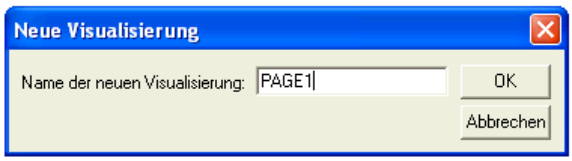
► In CoDeSys den Reiter [Visualisierungen] klicken.

► Neben dem Ordner-Symbol Rechtsklick auf [Visualisierungen], gefolgt von Klick auf [Objekt einfügen...]:



> Das Fenster [Neue Visualisierung] erscheint.

► Hinter [Name der neuen Visualisierung] in Großbuchstaben (!) den Namen des ersten Bildes eintragen (max. 8 Zeichen, keine Leerzeichen!):



► Mit [OK] übernehmen.

> CoDeSys öffnet das Zeichenfeld für diese Visualisierung:

HINWEIS: Das Zeichenfeld entspricht der Größe des Displays.

Zum Umgang mit dem Visualisierungs-Editor:
→ CoDeSys-Onlinehilfe oder
→ CoDeSys-Programmierhandbuch → ecomatmobile-DVD "Software, tools and documentation".

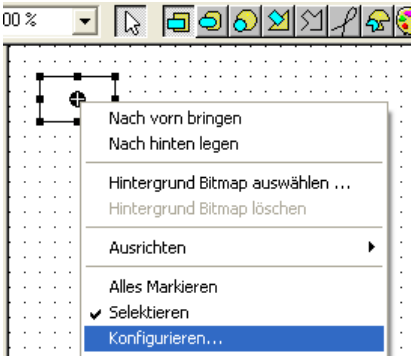
Zur Komplettierung unseres Test-Programms erstellen wir nun eine einfache Darstellung.

► Das Symbol "Rechteck" markieren.

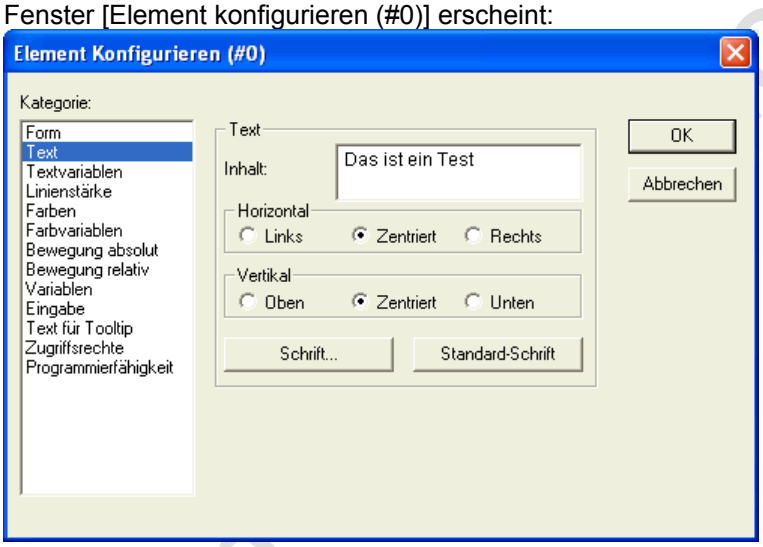
© ifm electronic gmbh

► Auf der Zeichenfläche auf einen Punkt zeigen als Beginn eines Rechtecks.
Linke Maustaste drücken und festhalten, dabei ein Rechteck in beliebiger Richtung aufziehen.
Am Endpunkt des Rechtecks die Maustaste wieder loslassen:

► Mit Rechtsklick auf das Rechteck Kontextmenü öffnen und [Konfigurieren...] wählen:



> Fenster [Element konfigurieren (#0)] erscheint:



► Im Feld [Kategorie] den Eintrag [Text] markieren.

► Im Feld [Text] > [Inhalt] einen Anzeigetext eintragen (→ Bild oben).

► Eintrag mit [OK] übernehmen.

► Projekt zwischendurch mit [Strg]+[s] sichern!

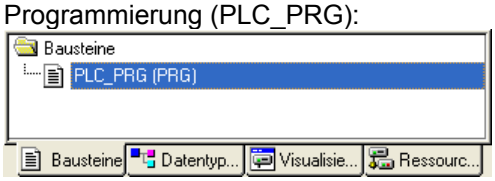
© ifm electronic gmbh

4.5.3 PLC-Programm erstellen

3101

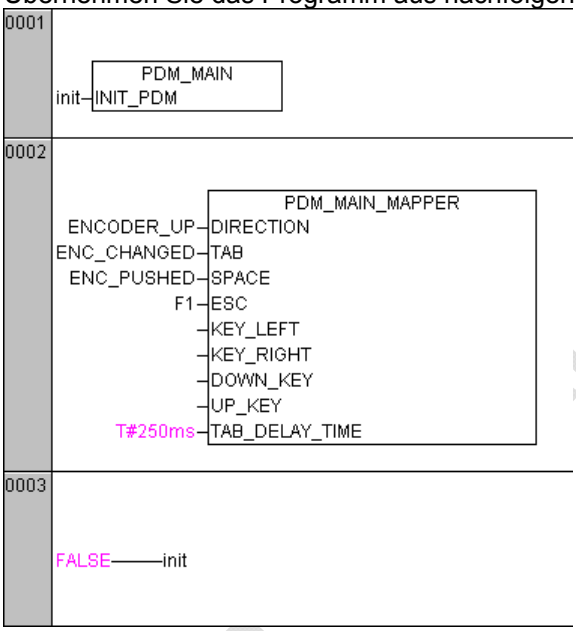
Für dieses Beispiel erstellen wir zuerst die Visualisierung, erst anschließend das PLC-Programm dazu.

► Für die eigentliche Programmierung wechseln Sie nun über den Reiter [Bausteine] wieder in die Programmierung (PLC_PRG):



Für ein lauffähiges Programm sind nur wenige Netzwerke erforderlich. Sie benötigen lediglich PDM_MAIN und PDM_MAIN_MAPPER aus der Bibliothek `ifm_CRnnnn_Vxxyyzz.LIB`, um wesentliche Funktionalitäten des Gerätes nutzen zu können.

► Übernehmen Sie das Programm aus nachfolgendem Beispiel:



> Sie können jetzt bereits Folgendes nutzen:
- den Tastenstatus abfragen,
- die LEDs setzen oder
- z.B. den Drehgeber abfragen.

Die Variable Init wird bei der Definition bereits auf TRUE gesetzt:

```
init: BOOL := TRUE
```

► Am Ende des ersten Zyklus müssen Sie die Variable `init` wieder zurücksetzen:
→ Netzwerk 3 im obigen Beispiel.

Alle wichtigen Systemvariablen für das PDM360, wie z.B. Taste `F1`, finden Sie hier:
→ unter dem Reiter [Ressourcen] oben in der Liste:
—→ Bibliothek `ifm_CRnnnn_Vxxyyzz.LIB`
——→ Globale Variablen <R> und
——→ PDM_MAIN <R>

© ifm electronic gmbh

4.5.4 Projekt im Gerät speichern

3046

Für folgende Geräte verfügbar:
- Controller CRnn32, CRnn33
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

► Im CoDeSys-Menü mit [Projekt] > [Alles Übersetzen] den Programmcode für das Gerät erzeugen.

> Wenn alle Bibliotheken und Variable wie beschrieben eingefügt und beschrieben sind, sollte es keine Fehlermeldungen geben.

► Wurde ein Projekt geändert (oder neu erstellt), muss es bereinigt werden, bevor es in das Gerät übertragen werden darf:
Menü [Projekt] > [Alles Bereinigen]

> Beim Bereinigen werden alle Programm-Elemente neu organisiert und zusammengefasst und somit der Umfang gemindert.
Ohne Bereinigen kann wegen Speicherüberlauf das Programm instabil werden.

► Diesen Stand im Computer mit [Strg]+[s] sichern.

► Mit Menü [Online] > [Kommunikationsparameter...] eine geeignete Schnittstelle definieren:
- Ethernet-Schnittstelle → *CoDeSys-Kommunikationsparameter für die Ethernet-Schnittstelle einstellen*, → Seite 34
- serielle Schnittstelle → *CoDeSys-Kommunikationsparameter für die serielle Schnittstelle einstellen*, → Seite 35
(nicht für PDM360NG: CR108n)
- CAN-Schnittstelle → *CoDeSys-Kommunikationsparameter für die CAN-Schnittstelle einstellen*
(nicht für PDM360NG: CR108n, PDM360compact: CR1052, CR1053, CR1055, CR1056)

► Passende Kabelverbindung Computer – PDM entsprechend der Parametrierung anschließen.

► Im CoDeSys-Menü mit [Online] > [Einloggen] eine logische Verbindung zum Gerät aufbauen.

> Meldung erscheint:
"Das Programm wurde geändert. Soll das neue Programm geladen werden?"

► Mit [Ja] bestätigen.

> Der Programmcode wird (zusammen mit der Visualisierung) auf das Gerät übertragen.

► Mit Menü [Online] > [Bootprojekt erzeugen] im Gerät eine Startdatei (= Bootprojekt) anlegen, so dass das Gerät nach einem Neustart mit dem aktuellen Programm startet. Andernfalls:
- das Gerät würde ohne Projekt starten (alle Geräte) oder (sofern vorhanden):
- das Gerät startet mit einer alten Projektversion (nicht bei PDM360NG: CR108n).

© ifm electronic gmbh

4.6 Geräte-Update auf neue Software-Version

Inhalt:

Geräte-Update vorbereiten....63
Geräte-Update durchführen 66

11548
3084

Immer, wenn es zu wesentlichen Verbesserungen in der Betriebsystem-Software oder des CoDeSys-Laufzeitsystems kommt, gibt **ifm** davon eine neue Version heraus. Die Versionen werden fortlaufend durchnummeriert (V01, V02, V03, ...).

Welche neuen Zusatzfunktionen die neue Softwareversion enthält, entnehmen Sie bitte der jeweiligen Dokumentation. Beachten Sie, ob in der Dokumentation auf besondere Anforderungen an die Hardware-Version hingewiesen wird.

Wenn Sie im Besitz eines Gerätes mit einer älteren Version sind und wenn die Bedingungen für die Hardware und Ihr Projekt stimmen, können Sie Ihr Gerät durch Aktualisieren der Software auf den neuen Software-Stand bringen.

ACHTUNG

Bei nicht exaktem Verlauf kann das Gerät anschließend nicht mehr booten.

► Wir empfehlen:
Wenn das Gerät ein Update erfordert, dann das Gerät über Ihren **ifm**-Fachvertrieb einsenden.

4.6.1 Geräte-Update vorbereiten

9829

Was wird benötigt?

9831

Was wird benötigt?	Woher?
aktuelle CoDeSys-Version	z.B. ecomat*mobile*-DVD "Software, tools and documentation"
TFTP Server-Programm	z.B. das Programm PumpKIN, → http://kin.klever.net/pumpkin
aktuelle Dateien des Software-Updates	• ecomat*mobile*-DVD "Software, tools and documentation" • ifm-Downloadbereich → www.ifm.com > Land wählen > [Service] > [Download] > [Steuerungssysteme]

Update-Dateien vorbereiten

9832

► Alle Dateien des Software-Updates auf den PC kopieren:
- Wenn noch nicht geschehen: ZIP-Archiv entpacken
- Dateien in ein beliebiges, neu angelegtes Verzeichnis speichern, z.B.:
`D:\PDMcompact\V04`

© ifm electronic gmbh

IP-Adresse des Server-PC einstellen

9833

Damit das Gerät eine Verbindung mit dem Server-PC aufbauen kann, muss dieser auf die IP-Adresse: 192.168.82.200 eingestellt sein.

Diese IP-Adresse ist zur Zeit fest und kann nicht geändert werden.

Andere Geräte im Netzwerk dürfen diese IP-Adresse NICHT haben, um Netzwerkstörungen zu vermeiden.

TFTP-Programm PumpKIN installieren und konfigurieren

9834

Das Programm ist ein kleines Tool, welches wenig PC-Ressourcen verbraucht.
Bei der Installation sind keine Administrator-Rechte erforderlich.

Download-Quelle z.B. → http://kin.klever.net/pumpkin

Diese Anleitung basiert auf der Version 2.7.1.

► Das Programm in ein beliebiges Verzeichnis installieren. ODER:
Die Setup-Voreinstellungen benutzen.

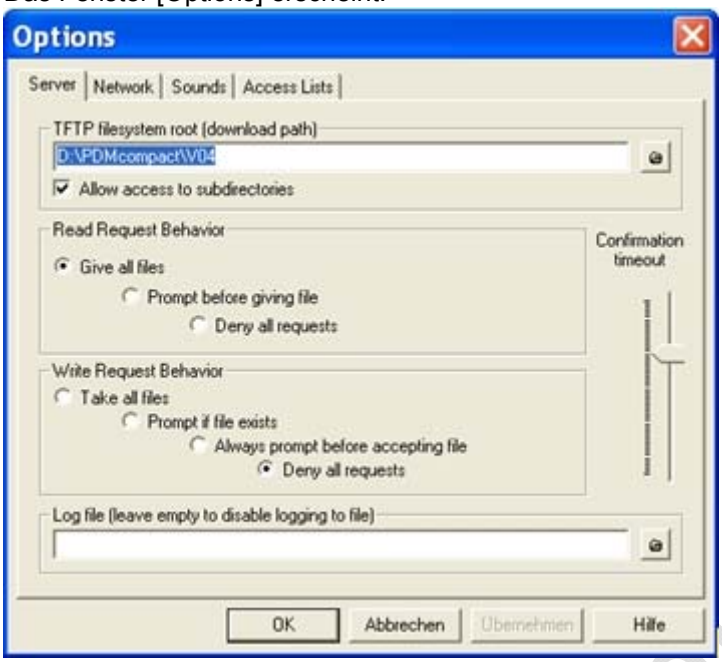
► Nach der Installation das Programm starten.

> Das Programm zeigt das Startbild:

© ifm electronic gmbh

► Die folgend beschriebenen Einstellungen vornehmen.
Dazu auf die Schaltfläche [Options] klicken.

> Das Fenster [Options] erscheint:



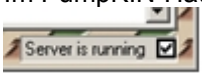
► Im Feld [TFTP filesystem root (download path)] das Verzeichnis angeben, in dem Sie ihre Update-Dateien gespeichert haben (z.B. `D:\PDMcompact\V04`).

► Im Feld [Read Request Behavior] das Optionsfeld [Give all files] aktivieren.

► Im Feld [Write Request Behavior] das Optionsfeld [Deny all request] aktivieren.

► Mit [OK] alle Einstellungen bestätigen und das Fenster schließen.

> Im PumpKIN-Hauptfenster sollte das Kontrollfeld [Server is running] aktiviert sein:



> Damit sind alle Einstellungen abgeschlossen.

© ifm electronic gmbh

4.6.2 Geräte-Update durchführen

9838

Ethernet-Verbindung und Spannungsversorgung herstellen

9839

► Eine Ethernet-Verbindung herstellen zwischen dem PC und dem PDM:

Verbindung von Computer und Gerät über ein vorhandenes Netzwerk (LAN):
8-poliges Verbindungskabel, M12-Stecker – RJ45-Stecker:

Länge	Artikel-Nr.	Foto
2 m	E11898	
10 m	E12204	
20 m	E12205	

Für den Anschluss im Netzwerk sollte ein erfahrener Anwender oder ein System-Administrator die Einrichtung der Netzwerkadressen und die Konfiguration vornehmen.

► Die Versorgungsspannung einschalten.
GLEICHZEITIG die Tasten [F1 esc]+[F2]+[F3] drücken und festhalten.

> Das Gerät schaltet in den Update-Modus.

► Die Tasten wieder loslassen.

© ifm electronic gmbh

Update-Software in das Gerät laden

9840

► Auf dem PC das Programm PumpKIN starten.

► Im Programm PumpKIN den Server starten.

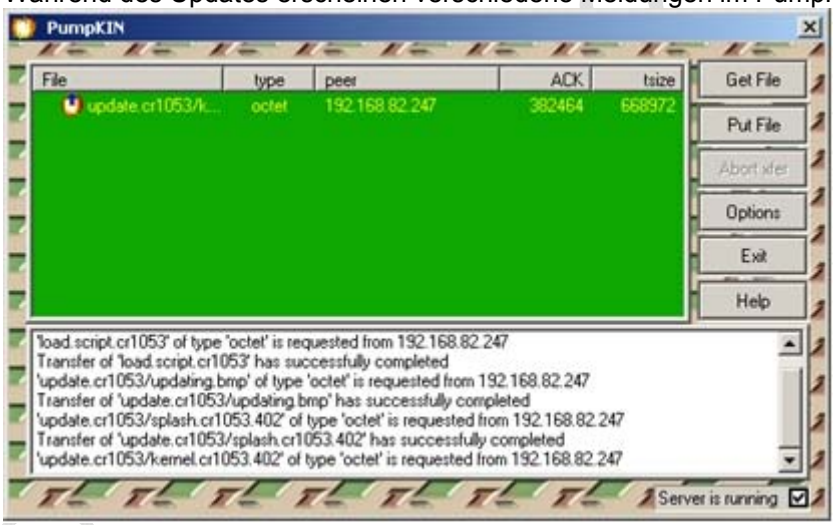
> Im PumpKIN-Hauptfenster sollte das Kontrollfeld [Server is running] aktiviert sein:

> Im PDM-Display erscheint nebenstehende Meldung.

> Der Update-Vorgang wurde damit automatisch gestartet.

WICHTIG:
Während des Updates die Versorgungsspannung auf keinen Fall unterbrechen!!

> Während des Updates erscheinen verschiedene Meldungen im PumpKIN-Fester, z.B.:



© ifm electronic gmbh

> Das Update dauert einige Minuten.

> Im PDM-Display erscheint anschließend nebenstehende Meldung.

► Mit [Exit] im TFTP-Programm das Programm PumpKIN beenden:

> Der Server-Modus des PC wird deaktiviert.

Die Ethernet-Verbindung wird nicht mehr benötigt und kann entfernt werden.

► Das PDM starten (Versorgungsspannung Aus/Ein).

> Das Startbild zeigt die neue Versionsnummer der Geräte-Software.

> Nach dem Booten startet das PDM-Setup-Programm.
Auch das Setup-Programm zeigt in der Statuszeile die neue Versionsnummer.

> Das Geräte-Update ist erfolgreich abgeschlossen.

© ifm electronic gmbh

5 Begrenzungen und Programmierhinweise

Inhalt:

Leistungsgrenzen des Geräts 69
Programmierhinweise für CoDeSys-Projekte.....75

3055

Hier zeigen wir Ihnen die Grenzen des Geräts und helfen Ihnen mit Programmierhinweisen.

5.1 Leistungsgrenzen des Geräts

7358

Beachten Sie die Grenzen des Geräts! → Datenblatt

5.1.1 CPU-Frequenzen

8005

► Beachten Sie, welche CPU in dem eingesetzten Gerät verwendet wird:

Controller-Familie / Artikel-Nr.	CPU-Frequenz [MHz]
BasicController: CR040n	50
CabinetController: CR0301, CR0302	20
CabinetController: CR0303	40
ClassicController: CR0020, CR0505	40
ClassicController: CR0032, CR0033	150
ExtendedController: CR0200	40
ExtendedController: CR0232, CR0233	150
SafetyController: CR7020, CR7021, CR7200, CR7201, CR7505, CR7506	40
SmartController: CR25nn	20

Monitor-Familie / Artikel-Nr.	CPU-Frequenz [MHz]
BasicDisplay: CR0451	50
PDM360: CR1050, CR1051	50
PDM360compact: CR1052, CR1053, CR1055, CR1056	50
PDM360NG: CR108n	400
PDM360smart: CR1070, CR1071	20

Je höher die CPU-Frequenz, desto größer ist die Leistungsfähigkeit für den gleichzeitigen Einsatz von komplexen Bausteinen.

© ifm electronic gmbh

5.1.2 Verhalten des Watchdog

1490

Bei (fast) allen programmierbaren **ecomat*mobile***-Geräten wird CoDeSys-intern die Programmlaufzeit über einen Watchdog überwacht.

Wird die maximale Watchdog-Zeit überschritten:
- das Gerät führt einen Reset durch und startet neu

JEDOCH:

BasicController: CR040n:
- alle Prozesse werden angehalten (Reset)
- alle Ausgänge werden ausgeschaltet
- die Status-LED blinkt rot mit 10 Hz
- Reset (nach Beheben des Fehlers) via Maintenance-Tool erforderlich

BasicDisplay: CR0451:
- alle Prozesse werden angehalten (Reset)
- alle Ausgänge werden ausgeschaltet
- der Bildschirm wird dunkel
- die Status-LED blinkt rot mit 10 Hz
- Reset (nach Beheben des Fehlers) via Maintenance-Tool erforderlich

SafetyController: CR7nnn:
- alle Prozesse werden angehalten (Reset)
- alle Ausgänge werden ausgeschaltet
- die Status-LED erlischt
- Neustart (nach Beheben des Fehlers) über Spannung Aus/Ein erforderlich

PDM360NG: CR108n:
- alle Prozesse werden angehalten (Reset)
- alle Ausgänge werden ausgeschaltet
- der Bildschirm wird dunkel
- die Status-LED blinkt rot mit 5 Hz
- Reset (nach Beheben des Fehlers) über Spannung Aus/Ein plus *Geräteparameter einstellen (Setup)* (→ Seite 11) erforderlich

Je nach Hardware haben die einzelnen Geräte ein unterschiedliches Zeitverhalten:

Gerät			Watchdog [ms]
BasicController: CR040n			100
BasicDisplay: CR0451 (Applikations-Programm) BasicDisplay: CR0451 (Visualisierung)			100 1.200
CabinetController: CR030n			100…200
ClassicController: CR0020, CR0032, CR0033, CR0505			100
ExtendedController: CR0200, CR0232, CR0233			100
Platinensteuerung: CS0015			100…200
SafetyController: CR7nnn			100
SmartController: CR25nn			100…200
PDM360: CR1050, CR1051			kein Watchdog
PDM360compact: CR1052, CR1053, CR1055, CR1056			kein Watchdog
PDM360NG: CR108n	Prozess	Chain ID [dez]	
Watchdog-Reset *) = 500 ms	ppc-plclinux	10	1 000
	eplc main	20	10 000
	eplcoverlay	21	10 000
	keyboard-service	40	2 000
	io-coproc-service	50	2 000
	wd-coproc-service	60	2 000
PDM360smart: CR1070, CR1071			100…200

*) Watchdog-Reset ist das Intervall, mit dem sich die Prozesse beim Watchdog-Prozessor melden.

© ifm electronic gmbh

5.1.3 Begrenzungen beim PDM360compact

9847

Zu den Grenzwerten unbedingt das Datenblatt beachten!

Beachten Sie insbesondere folgende Begrenzungen:

Bezeichnung	PDM360compact CR1052, CR1053	PDM360compact CR1055, CR1056
Länge Strings	≤ 80 Zeichen	≤ 80 Zeichen
Länge Pfadnamen	≤ 80 Zeichen	≤ 80 Zeichen
Anzahl grafische Objekte pro Visualisierungsseite	50...100	50...100
Anzahl Bitmaps ¹) pro Projekt	≤ 100	≤ 100
Anzahl Zeichensätze pro Projekt	≤ 5	≤ 5
Anzahl POUs ²) pro Projekt	≤ 24.576	≤ 24.576

¹) Vorgaben für das Startbild → Kapitel *Startbild des PDM ändern*, → Seite 17.

²) POU (Program Organization Unit) = Funktion, Funktionsblock oder Programmblock

Weitere Begrenzungen für Programm- und Datenspeicher:

Bezeichnung	PDM360compact CR1052, CR1053	PDM360compact CR1055, CR1056
Speicher gesamt	16 MByte RAM 8 MByte Flash	16 MByte RAM 8 MByte Flash
davon für Programm-Code mit Online-Change ¹)	1,8 MByte	1,8 MByte
ODER für Programm-Code ohne Online-Change ¹)	3,0 MByte	3,0 MByte
Voreinstellung Online-Change ¹)	nicht möglich	nicht möglich
Datenspeicher	0,5 MByte	0,5 MByte
Speicher für Retain-Daten ²)	1 kByte	1 kByte
Speicher im Verzeichnis `/tmp`	keine Zusicherung	keine Zusicherung

¹) Online-Change = Möglichkeit, online den Programm-Code zu ändern, ohne zuvor das ursprüngliche Programm löschen zu müssen.

²) Retain-Daten → *Retain-Variable*, → Seite 81

© ifm electronic gmbh

5.1.4 Visualisierungsgrenzen

Inhalt:

Bild umrechnen / skalieren 73
CoDeSys-Visualisierungs-Elemente 74
Texte 75

3114

Embeded-Displays, wie sie z.B. im PDM360 verbaut sind, können den vollen Farbumfang von Bitmap-Grafiken nicht zur Verfügung stellen, weil nur eingeschränkte Leistungsreserven verfügbar sind. Folgende Vorbereitungen ermöglichen trotzdem Bitmap-Bilder im PDM:

- richtige Auswahl der Motive,
- geschicktes Verschieben von Farben oder
  geschickte Zusammenstellung einer Farbtabelle und
- das richtige Skalieren der Bitmaps vor dem Einsatz auf dem PDM.

Leistungsreserven des Geräts → Kapitel *Leistungsgrenzen des Geräts*, → Seite 69

Tabelle: Bild-Vorgaben für das Startbild

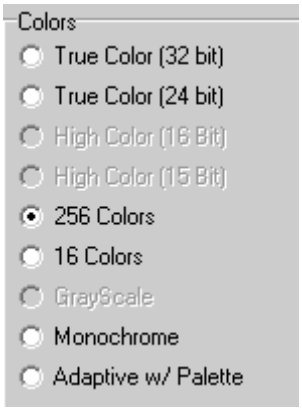
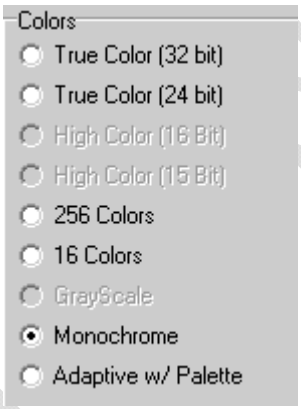
	CR1051	CR1050	CR1052, CR1053	CR1055, CR1056
Datei-Typ	Bitmap (`*.bmp`)			
Dateiname	nur Kleinbuchstaben, Namenskonvention = 8.3			
Bildgröße	320x240 Pixel	320x240 Pixel	240x320 Pixel	320x240 Pixel
Farben	8 Bit = 256 Farben	1 Bit = nur Schwarz und Weiß, keine Graustufen		
Speicherbedarf	ca. 77 kByte	ca. 10 kByte		

→ *Startbild des PDM ändern*, → Seite 17

Die im Projekt eingesetzten Grafiken dürfen durchaus größer sein als die vorgegebene Bildgröße. Dann wird von dem Bild jedoch nur ein (wählbarer) Ausschnitt sichtbar sein.

© ifm electronic gmbh

Farben:

PDM color	PDM monochrome
CR1051	CR1050, CR1052, CR1053, CR1055, CR1056
unterstützt 256 Farben aus einem gesamten Farbraum von 512 Farben	unterstützt nur die 2 Farben Schwarz und Weiß
Bitmap mit 8 Bit Farbtiefe	Monochrom-Bitmap Bilevel
Beim Farb-Bitmap kann die gesamte Farbpalette aus 2^8 = 256 Farben genutzt werden.	Beim Monochrom-Bitmap sollten nur die Farben Weiß (R=0, G=0, B=0) und Schwarz (R=224, G=224, B=224) oder das Monochrom-Farbformat Bilevel verwendet werden.
	

→ *Welche Farben werden dargestellt?*

Bild umrechnen / skalieren

3117

Wird im Gerät ein Bild geladen, welches nicht den Größen- oder den Farbanforderungen genügt, wird es vor der Anzeige auf die richtige Größe skaliert und die verwendeten Farben "überprüft".

Dieses Umrechnen eines Bildes muss bei jedem Aufruf des Bildes erneut durchgeführt werden. So verlängern sich oft die Bildwechsel-Zeiten erheblich. Abhilfe:

► Alle Umformungen des Bitmaps oder des Bildes zuvor auf dem Computer in einer Bildverarbeitung durchführen.
Nur für BasicDisplay: CR0451: Die Anpassung an die *Farbpalette* erfolgt beim Einbinden des Bildes im Projekt durch CoDeSys. Auf dem Gerät selbst werden keine Anpassungen vorgenommen (Größe, Skalierung, Farbe).

► Nur die passend gewandelten Bilder in der Visualisierung des Geräts speichern.

→ Kapitel *Bildgröße Vektorgrafik / Pixelgrafik*, → Seite 320

© ifm electronic gmbh

CoDeSys-Visualisierungs-Elemente

9849

Gilt nur für folgende Geräte:
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Nicht alle CoDeSys-Funktionen können auf dem PDM erfolgreich arbeiten:

Visualisierungselement	Funktions-Sicherheit beim PDM	
Rechteck	+	keine Probleme bekannt
abgerundetes Rechteck	—	wird nicht unterstützt
Kreis, Ellipse	+	keine Probleme bekannt
Polygone	o	möglich, jedoch zu viele Elemente davon auf einer Seite bremsen das System
Tortengrafik	o	nur sehr eingeschränkt verwendbar; bremsen das System sehr stark
BMP-Grafikdateien	+	≤ 100 pro Projekt
Visualisierung	o	möglich, jedoch zu viele Elemente davon auf einer Seite bremsen das System
Schaltflächen	+	keine Probleme bekannt
WMF-Grafikdateien	—	wird nicht unterstützt
Tabellen	—	nicht sinnvoll nutzbar
Trendkurven	o	nur sehr eingeschränkt verwendbar; bremsen das System sehr stark
Alarmtabelle	—	nicht sinnvoll nutzbar
Skalen	o	möglich, jedoch zu viele Elemente davon auf einer Seite bremsen das System. → nachfolgenden Hinweis.
Balkendiagramm	+	keine Probleme bekannt
Histogramm	+	keine Probleme bekannt

Zum Vermeiden zu langer Bild-Ladezeiten beachten Sie bitte:

- In der Grafik grafische Elemente nicht gruppieren!
- Grafiken möglichst nicht überlagern.
- Manche Visualisierungen mit den CoDeSys-Möglichkeiten sind nicht sehr befriedigend, z.B. runde Skalen. Abhilfe:
  Integrieren Sie die gewünschten Elemente als (extern erzeugte) BMP-Grafik. In der Visualisierung muss dann nur ein Pfeil werteabhängig gedreht werden, der bei Bedarf bei Überschreiten von Grenzwerten seine Farbe wechseln könnte.

© ifm electronic gmbh

Texte

9306

Gilt nur für folgende Geräte:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

- Zum Vermeiden zu langer Bild-Ladezeiten:
  Reduzieren Sie die Anzahl verschiedener Zeichensätze (Fonts) je Projekt.
- Die kleinste auf dem PDM gut lesbare Schriftgröße ist 8 Punkt.

5.2 Programmierhinweise für CoDeSys-Projekte

Inhalt:

FB, FUN, PRG in CoDeSys....76
Bibliotheken....77
Arbeitsreihenfolge....78
Speichern....78
ifm-Maintenance-Tool nutzen....79

7426

Hier erhalten Sie Tipps zum Programmieren des Geräts.

► Beachten Sie die Hinweise im CoDeSys-Programmierhandbuch
→ **ecomat*mobile***-DVD "Software, tools and documentation".

© ifm electronic gmbh

5.2.1 FB, FUN, PRG in CoDeSys

8473

In CoDeSys unterscheiden wir folgende Typen von Bausteinen (POUs):

FB = function block = Funktionsblock

- Ein FB kann mehrere Eingänge und mehrere Ausgänge haben.
- Ein FB darf in einem Projekt mehrmals aufgerufen werden.
- Für jeden Aufruf muss eine Instanz deklariert werden.
- Erlaubt: Im FB aufrufen von FB und FUN.

FUN = function = Funktion

- Eine Funktion kann mehrere Eingänge, aber nur einen Ausgang haben.
- Der Ausgang ist vom gleichen Datentyp wie die Funktion selbst.

PRG = program = Programm

- Ein PRG kann mehrere Eingänge und mehrere Ausgänge haben.
- Ein PRG darf in einem Projekt nur einmal aufgerufen werden.
- Erlaubt: im PRG aufrufen von PRG, FB und FUN.

HINWEIS

Funktionsblöcke dürfen NICHT in Funktionen aufgerufen werden.
Sonst: Bei der Ausführung stürzt das Applikations-Programm ab.

Alle Bausteine (POUs) dürfen NICHT rekursiv aufgerufen werden, auch nicht indirekt.

Hintergrund:

Alle Variablen von Funktionen...
- werden beim Aufruf initialisiert und
- werden nach der Rückkehr zum Aufrufer ungültig.

Funktionsbausteine haben 2 Aufrufe:
- einen Initialisierungsaufruf und
- den eigentlichen Aufruf, um irgend etwas zu tun.

Folglich heißt das für den FB-Aufruf in einer Funktion, dass jedesmal ein zusätzlicher Initialisierungsaufruf über die Schnittstelle ginge.

© ifm electronic gmbh

5.2.2 Bibliotheken

9939

Folgende Bibliotheken sollten die CoDeSys-Projekte mindestens enthalten:

- Standard-Bibliothek `Standard.Lib` in `C:\...\CoDeSys\Library\`
- Geräte-Bibliothek `ifm_CRnnnn_Vxxyyzz.LIB` in `C:\...\CoDeSys\Targets\ifm\Library\ifm_CRnnnn`

Bei Einsatz von PDM als CANopen-Master sind folgende Bibliotheken mindestens erforderlich:

- `3S_CanDrv.lib` in `C:\...\CoDeSys\Library\`
- `3S_CanOpenNetVar.lib` in `C:\...\CoDeSys\Library\`
- `3S_CanOpenManager.lib` in `C:\...\CoDeSys\Library\`
- `3S_CanOpenMaster.lib` in `C:\...\CoDeSys\Library\`

Bei Einsatz von PDM als CANopen-Slave sind folgende Bibliotheken mindestens erforderlich:

- `3S_CanDrv.lib` in `C:\...\CoDeSys\Library\`
- `3S_CanOpenNetVar.lib` in `C:\...\CoDeSys\Library\`
- `3S_CanOpenManager.lib` in `C:\...\CoDeSys\Library\`
- `3S_CanOpenDevice.lib` in `C:\...\CoDeSys\Library\`

Zur Behandlung von Retain-Variablen:

- SysSaveRetain und SysRestoreRetain (aus der Bibliothek `SysLibPlcCtrl.Lib` in `C:\...\CoDeSys\Library\`)

Zur Behandlung von Dateien und zum Mitschreiben von Daten:
Gefahr für das System bei falscher Handhabung! Erfahrung erforderlich!

- Bibliothek `SysLibFile.Lib` in `C:\...\CoDeSys\Library\` **ODER:**
- Bibliothek `ifm_PDM_FILE_Vxxyyzz.LIB` in `C:\...\CoDeSys\Targets\ifm\Library\`

© ifm electronic gmbh

5.2.3 Arbeitsreihenfolge

7427

Es gibt grundsätzlich zwei Reihenfolgen, ein Projekt für ein PDM oder Display zu erstellen:

A) Zuerst die Visualisierung, anschließend das PLC-Programm.	
Vorteile: • Im Programm kann auf die Parameter in den fertigen Bildern querverwiesen werden. • Beim Testen des PLC-Programms existieren die Bilder bereits.	**Nachteil:** • Die in den Bildern benötigten PLC-Parameter und Variablen sind noch nicht definiert.
B) Zuerst das PLC-Programm, anschließend die Visualisierung.	
Vorteil: • Alle Parameter und Variablen sind im PLC-Programm definiert, bevor in den Visualisierungen auf sie verwiesen wird.	**Nachteile:** • Die Parameter aus den Bildern (Bildnummer, Taste, LED usw.) müssen anderweitig ermittelt werden. • Das PLC-Programm kann erst nach dem Erstellen der Visualisierung getestet werden.

In beiden Fällen empfehlen wir dringend, **vor** Beginn eine möglichst genaue Struktur der Visualisierung und ihrer Inhalte zu entwerfen.

5.2.4 Speichern

7430

Gilt nur für folgende Geräte:
- Controller CRnn32, CRnn33
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Nur Dateien im Flash-Speicher (oder im EEPROM) sind vor Spannungsausfall geschützt.

Speichern Sie im Gerät zusammen mit Ihrem CoDeSys-Projekt immer auch das zugehörige Boot-Projekt:

► Menü [Online] > [Bootprojekt erzeugen] (dies muss nach jeder Änderung erneut erfolgen!).

> Nach einem Neustart startet das Gerät mit dem zuletzt gespeicherten Bootprojekt.

© ifm electronic gmbh

5.2.5 ifm-Maintenance-Tool nutzen

8492

Für folgende Geräte verfügbar:
- BasicController: CR040n
- BasicDisplay: CR0451
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Das **ifm**-Maintenance-Tool dient dem einfachen Übertragen des Programmcodes vom Programmierplatz in das Gerät. Grundsätzlich kann jede Applikations-Software mit dem **ifm**-Maintenance-Tool auf die Geräte kopiert werden. Vorteil: Dazu ist kein Programmiersystem mit einer CoDeSys-Lizenz erforderlich.

Sicherheitsrelevante Applikations-Software MUSS mit dem **ifm**-Maintenance-Tool auf die Steuerungen kopiert werden, um die Prüfsumme CRC, mit der die Software zertifiziert wurde, nicht zu verfälschen.

© ifm electronic gmbh

6 Variable

Inhalt:

Retain-Variable 81
Netzwerkvariablen 82

3130

In diesem Kapitel erfahren Sie mehr über den Umgang mit Variablen.

Grundsätzlich unterscheiden wir in CoDeSys folgende Arten von Variablen:

Variable	Deklaration	Gültigkeitsbereich	Speicherverhalten
lokal	im einzelnen Baustein: PROGRAM... VAR (Deklaration) END_VAR	Gilt nur im Baustein (POU), wo sie konfiguriert wurde.	flüchtig
global	in [Ressourcen] > [Globale Variablen] > [Globale_Variablen]: VAR_GLOBAL (Deklaration) END_VAR	Gilt in allen Bausteinen (POUs) dieses Projekts.	
lokal Retain	im einzelnen Baustein: PROGRAM... VAR RETAIN (Deklaration) END_VAR	Gilt nur im Baustein (POU), wo sie konfiguriert wurde.	Werte werden (auf Anforderung) gesichert: im FRAM-Speicher [1]): - alle ecomat*mobile*-Controller, - BasicController: CR040n, - BasicDisplay: CR0451, - PDM360NG: CR108n - PDM360smart: CR1070, CR1071 in spezieller Datei: - PDM360: CR1050, CR1051 - PDM360compact: CR1052, CR1053, CR1055, CR1056
global Retain	in [Ressourcen] > [Globale Variablen] > [Globale_Variablen]: VAR_GLOBAL RETAIN (Deklaration) END_VAR	Gilt in allen Bausteinen (POUs) dieses CoDeSys-Projekts.	
Netzwerk	in [Ressourcen] > [Globale Variablen] > Deklarationsliste: VAR_GLOBAL (Deklaration) END_VAR	Werte stehen allen CoDeSys-Projekten im gesamten Netzwerk zur Verfügung, wenn die Variable in ihren Deklarationslisten enthalten ist.	flüchtig
Netzwerk Retain	in [Ressourcen] > [Globale Variablen] > Deklarationsliste: VAR_GLOBAL RETAIN (Deklaration) END_VAR		Werte werden (auf Anforderung) gesichert: im FRAM-Speicher [1]): - alle ecomat*mobile*-Controller, - BasicController: CR040n, - BasicDisplay: CR0451, - PDM360NG: CR108n - PDM360smart: CR1070, CR1071 in spezieller Datei: - PDM360: CR1050, CR1051 - PDM360compact: CR1052, CR1053, CR1055, CR1056

[1]) FRAM steht hier allgemein für alle Arten von nichtflüchtigen, schnellen Speichern.

Im Folgenden beschreiben wir die Besonderheiten im PDM für ...
- *Retain-Variable* (→ Seite 81) und
- *Netzwerkvariablen* (→ Seite 82).

→ CoDeSys-Programmierhandbuch → **ecomat*mobile***-DVD "Software, tools and documentation"

© ifm electronic gmbh

6.1 Retain-Variable

8672

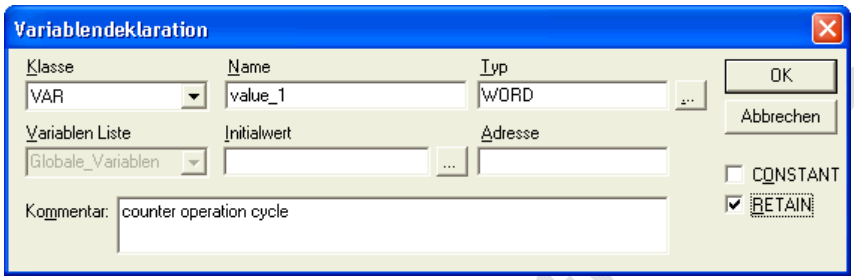
Retain-Variable werden automatisch in einen geschützten Speicherbereich gesichert und automatisch bei Neustart wieder geladen.

Typische Einsätze für Retain-Variable sind z.B.:

- Betriebsstunden, die zur Laufzeit der Maschine fortgeschrieben werden,
- Positionswerte von Inkrementalgebern,
- im PDM eingetragene Sollwerte,
- Maschinenparameter,
- also alle Variablen, deren Werte beim Ausschalten des Geräts nicht verloren gehen dürfen.

Als Retain können alle Variablentypen, auch komplexe Stukturen (z.B. Timer), gekennzeichnet werden.

► Dazu in der Variablendeklaration das Kontrollfeld [RETAIN] aktivieren (→ Bild).



6.1.1 Sichern von Retain-Variablen

9853

Im Gerät werden auch die Daten vom Typ RETAIN zur Laufzeit nur im flüchtigen Speicher (RAM) abgelegt.
Um sie dauerhaft zu sichern, werden sie am Ende jedes Zyklus automatisch in den FRAM-Speicher [1]) geschrieben.

[1]) FRAM steht hier allgemein für alle Arten von nichtflüchtigen, schnellen Speichern.

HINWEIS

In diesem Gerät die folgenden Funktionen aus der 3S-Bibliothek `SysLibPlcCtrl.lib` NICHT einsetzen:
- FUN SysSaveRetains
- FUN SysRestoreRetains

© ifm electronic gmbh

6.1.2 Rücklesen von Retain-Variablen

9854

Nach dem Einschalten und vor dem ersten Programmzyklus schreibt das Gerät die gesicherten Daten einmalig automatisch zurück in den Arbeitsspeicher. Dazu müssen keine zusätzlichen Bausteine in das Applikations-Programm integriert werden.

HINWEIS

In diesem Gerät die folgenden Funktionen aus der 3S-Bibliothek `SysLibPlcCtrl.lib` NICHT einsetzen:
- FUN SysSaveRetains
- FUN SysRestoreRetains

6.2 Netzwerkvariablen

9856

→ Kapitel *CANopen-Netzwerkvariablen*, → Seite 135

© ifm electronic gmbh

7 CAN einsetzen

Inhalt:

Allgemeines zu CAN 83
Physikalische Anbindung des CAN 88
CAN-Datenaustausch 92
Beschreibung der CAN-Standardbausteine 96
ifm-CANopen-Bibliotheken 101
CAN-Fehler und Fehlerbehandlung 170

3043

7.1 Allgemeines zu CAN

Inhalt:

Topologie 84
CAN-Schnittstellen 85
Verfügbare CAN-Schnittstellen und CAN-Protokolle 85
System-Konfiguration 87
Software für CAN und CANopen 87

1164

Der CAN-Bus (**C**ontroller **A**rea **N**etwork) gehört zu den Feldbussen.

Es handelt sich dabei um ein asynchrones, serielles Bussystem, das 1983 von Bosch für die Vernetzung von Steuergeräten in Automobilen entwickelt und 1985 zusammen mit Intel vorgestellt wurde, um die Kabelbäume (bis zu 2 km pro Fahrzeug) zu reduzieren und dadurch Gewicht zu sparen.

© ifm electronic gmbh

7.1.1 Topologie

1244

Das CAN-Netzwerk wird als Linienstruktur aufgebaut. Stichleitungen sind in eingeschränktem Umfang zulässig. Des Weiteren sind auch ein ringförmiger Bus (Infotainment Bereich) sowie ein sternförmiger Bus (Zentralverrieglung) möglich. Beide Varianten haben im Vergleich zum linienförmigen Bus jeweils einen Nachteil:

- Im ringförmigen Bus sind alle Steuergeräte in Reihe geschaltet, so dass bei einem Ausfall eines Steuergeräts der gesamte Bus ausfällt.
- Der sternförmige Bus wird meist von einem Zentralrechner gesteuert, da diesen alle Informationen passieren müssen, mit der Folge, dass bei einem Ausfall des Zentralrechners keine Informationen weitergeleitet werden können. Bei einem Ausfall eines einzelnen Steuergeräts funktioniert der Bus weiter.

Der lineare Bus hat den Vorteil, dass alle Steuergeräte parallel zu einer zentralen Leitung gehen. Nur wenn diese ausfällt, funktioniert der Bus nicht mehr.

HINWEIS

► Die Linie muss an ihren beiden Enden jeweils mit einem Abschlusswiderstand von der Größe 120 Ohm abgeschlossen werden, um ein Verfälschen der Signalqualität zu verhindern.

Die Geräte der **ifm electronic gmbh**, die mit einem CAN-Interface ausgestattet sind, haben grundsätzlich keine Abschlusswiderstände.

Stichleitungen und sternförmiger Bus haben den Nachteil, dass der Wellenwiderstand schwer zu bestimmen ist. Im schlimmsten Fall funktioniert der Bus nicht mehr.

© ifm electronic gmbh

7.1.2 CAN-Schnittstellen

2269

Die Controller werden je nach Aufbau der Hardware mit mehreren CAN-Schnittstellen ausgerüstet. Grundsätzlich können alle Schnittstellen unabhängig voneinander mit folgenden Funktionen genutzt werden:

- Layer 2: CAN auf Ebene 2
- CANopen (→ Kapitel *ifm-CANopen-Bibliotheken*, → Seite 101), ein Protokoll nach CiA 301/401 für Master/Slave-Betrieb (via CoDeSys)
- *CANopen-Netzwerkvariablen* (→ Seite 135) (via CoDeSys)
- Protokoll SAE J1939 (für Antriebs-Management, → Kapitel *CAN-Bausteine nach SAE J1939*)
- Buslast-Erkennung
- Errorframe-Zähler
- Download-Schnittstelle (nicht alle Geräte)
- 100 % Buslast ohne Paketverlust

Welche CAN-Schnittstelle des Geräts welche konkreten Möglichkeiten bietet, → Datenblatt des Geräts.
Das aktuelle Datenblatt finden Sie auf der **ifm**-Homepage:
→ www.ifm.com > Land wählen > [Datenblattsuche] > Artikel-Nr.

Weitere interessante CAN-Protokolle sind:

- "Truck & Trailer Interface" nach ISO 11992 (→ Kapitel *Nutzung der CAN-Schnittstelle nach ISO 11992*)
  Für folgende Geräte verfügbar: SmartController: CR2501
- ISOBUS nach ISO 11783 für Landmaschinen
- NMEA 2000 für den maritimen Einsatz
- CANopen Truck Gateway nach CiA 413 (Umsetzung zwischen ISO 11992 und SAE J1939)

7.1.3 Verfügbare CAN-Schnittstellen und CAN-Protokolle

6467

In den **ecomatmobile**-Geräten sind folgende CAN-Schnittstellen und CAN-Protokolle verfügbar:

Schnittstelle voreingestellter Download-ID **Gerät**	**CAN 1** ID 127	**CAN 2** ID 126	**CAN 3** ID 125	**CAN 4** ID 124	**Standard Baudrate** [kBit/s]
BasicController: CR040n	CAN Layer 2 CANopen SAE J1939	CAN Layer 2 CANopen SAE J1939	---	---	250
BasicDisplay: CR0451	CAN Layer 2 CANopen SAE J1939	---	---	---	250
CabinetController: CR0301, CR0302	CAN Layer 2 CANopen SAE J1939	---	---	---	125
CabinetController: CR0303	CAN Layer 2 CANopen SAE J1939	CAN Layer 2 SAE J1939	---	---	125

© ifm electronic gmbh

Schnittstelle voreingestellter Download-ID **Gerät**	**CAN 1** ID 127	**CAN 2** ID 126	**CAN 3** ID 125	**CAN 4** ID 124	**Standard Baudrate** [kBit/s]
ClassicController: CR0020, CR0505	CAN Layer 2 CANopen SAE J1939	CAN Layer 2 SAE J1939	---	---	125
ClassicController: CR0032, CR0033	CAN Layer 2 CANopen SAE J1939	CAN Layer 2 CANopen SAE J1939	CAN Layer 2 CANopen SAE J1939	CAN Layer 2 CANopen SAE J1939	125
ExtendedController: CR0200	CPU 1 CAN 1 ID 127 CAN Layer 2 CANopen SAE J1939	CPU 1 CAN 2 ID 126 CAN Layer 2 SAE J1939	CPU 2 CAN 1 ID 127 CAN Layer 2 CANopen SAE J1939	CPU 2 CAN 2 ID 126 CAN Layer 2 SAE J1939	125
ExtendedController: CR0232, CR0233	CAN Layer 2 CANopen SAE J1939	CAN Layer 2 CANopen SAE J1939	CAN Layer 2 CANopen SAE J1939	CAN Layer 2 CANopen SAE J1939	125
Platinensteuerung: CS0015	CAN Layer 2 CANopen SAE J1939	---	---	---	Drehschalter
SafetyController: CR7021, CR7506	CAN Layer 2 CANopen CANopen-Safety SAE J1939	CAN Layer 2 CANopen-Safety SAE J1939	---	---	125
ExtendedSafetyController: CR7201	CPU 1 CAN 1 ID 127 CAN Layer 2 CANopen CANopen-Safety SAE J1939	CPU 1 CAN 2 ID 126 CAN Layer 2 CANopen-Safety SAE J1939	CPU 2 CAN 1 ID 127 CAN Layer 2 CANopen SAE J1939	CPU 2 CAN 2 ID 126 CAN Layer 2 SAE J1939	125
SmartController: CR2500	CAN Layer 2 CANopen SAE J1939	CAN Layer 2 SAE J1939	---	---	125
PDM360: CR1050, CR1051	CAN Layer 2 CANopen	CAN Layer 2 CANopen SAE J1939	---	---	125
PDM360compact: CR1052, CR1053, CR1055, CR1056	CAN Layer 2 CANopen	---	---	---	125
PDM360smart: CR1070, CR1071	CAN Layer 2 CANopen SAE J1939	---	---	---	125
PDM360NG: CR108n	CAN Layer 2 CANopen SAE J1939	CAN Layer 2 CANopen SAE J1939	CAN Layer 2 CANopen SAE J1939	CAN Layer 2 CANopen SAE J1939	125

© ifm electronic gmbh

7.1.4 System-Konfiguration

8573
8572

Die **ecomatmobile**-Geräte werden mit folgenden Download-Identifier (= ID) ausgeliefert:
- ID 127 für CAN-Schnittstelle 1
- ID 126 für CAN-Schnittstelle 2 (wenn vorhanden)
- ID 125 für CAN-Schnittstelle 3 (wenn vorhanden)
- ID 124 für CAN-Schnittstelle 4 (wenn vorhanden)

Das Download-System benutzt diesen Identifier für die erste Kommunikation mit einem nicht konfigurierten Modul über CAN.
Die Download-IDs können auf folgenden Wegen eingestellt werden:
- über den PLC-Browser des Programmiersystems,
- über den Downloader oder das MaintenanceTool oder
- über das Applikations-Programm.

Über den Modus "Autoconfig" des Bootloaders kann nur die CAN-Schnittstelle 1 eingestellt werden.

7.1.5 Software für CAN und CANopen

1185

Grundsätzlich können **ecomatmobile**-Geräte durch Nutzung von *CAN_x_TRANSMIT* (→ Seite 99) und *CAN_x_RECEIVE* (→ Seite 97) direkt an der CAN-Kommunikation teilnehmen (Schicht 2). In der Betriebsart CANopen werden dem Programmierer die festgelegten Dienste aus dem Programmiersystem CoDeSys zur Verfügung gestellt.

Folgende Punkte sind zu beachten:

- In der Betriebsart "CAN-Direkt" auf Schicht 2 ist der Programmierer für alle Dienste selbst verantwortlich. In diesem Zustand befindet sich das Gerät nach folgenden Ereignissen:
  - nach einem Programm-Download oder
  - nach einem Reset-Kommando durch das Programmiersystem.
- Durch Einbinden der CoDeSys-CANopen-Systembibliotheken (Funktionen in den Zielsystemeinstellungen aktivieren) wird die Betriebsart CANopen aktiviert. Je nach gewählter Funktion läuft das Gerät als CANopen-Master oder -Slave (→ Kapitel *ifm-CANopen-Bibliotheken*, → Seite 101).

© ifm electronic gmbh

7.2 Physikalische Anbindung des CAN

Inhalt:

Netzaufbau 88
CAN-Buspegel 89
Busleitungslänge 90
Leitungsquerschnitte 91

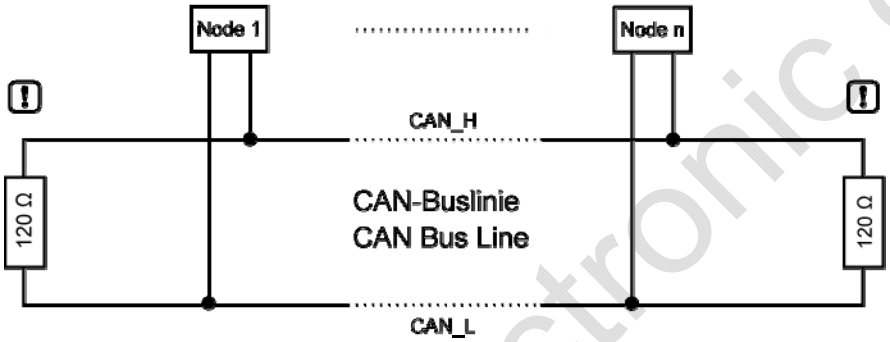
1177

Die in den Kapiteln *CAN-Datenaustausch* (→ Seite 92) und *CAN-Fehler und Fehlerbehandlung* (→ Seite 170) beschriebenen Mechanismen der Datenübertragung und der Fehlerbehandlung sind direkt im CAN-Controller implementiert. Die physikalische Verbindung der einzelnen CAN-Teilnehmer wird von der ISO 11898 in der Schicht 1 beschrieben.

7.2.1 Netzaufbau

1178

Die Norm ISO 11898 setzt einen Aufbau des CAN-Netzes mit einer Linienstruktur voraus.



Grafik: CAN-Netzaufbau Linienstruktur

HINWEIS

► Die Linie muss an ihren beiden Enden jeweils mit einem Abschlusswiderstand von der Größe 120 Ohm abgeschlossen werden, um ein Verfälschen der Signalqualität zu verhindern.

Die Geräte der **ifm electronic gmbh**, die mit einem CAN-Interface ausgestattet sind, haben grundsätzlich keine Abschlusswiderstände.

Stichleitungen

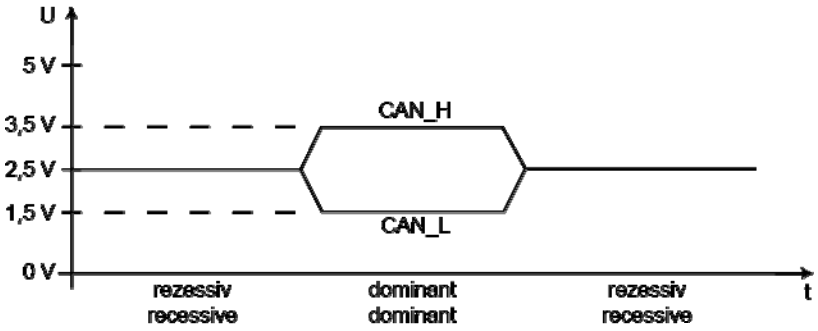
Idealerweise sollte zu den Busteilnehmern (Node 1 ... Node n) keine Stichleitung führen, da in Abhängigkeit von der Gesamtleitungslänge und den zeitlichen Abläufen auf dem Bus Reflektionen auftreten. Damit diese nicht zu Systemfehlern führen, sollten die Stichleitungen zu einem Busteilnehmer (z.B. einem E/A-Modul) eine gewisse Länge nicht überschreiten. Stichleitungen mit einer Länge von 2 m (bezogen auf 125 kBit/s) werden als unkritisch angesehen. Die Summe aller Stichleitungen im Gesamtsystem sollte 30 m nicht übersteigen. In besonderen Fällen müssen die Leitungslängen der Linie und der Stiche genau berechnet werden.

© ifm electronic gmbh

7.2.2 CAN-Buspegel

1179

Der CAN-Bus befindet sich im inaktiven (rezessiven) Zustand, wenn die Ausgangstransistorpaare in allen Busteilnehmern ausgeschaltet sind. Wird mindestens ein Transistorpaar eingeschaltet, wird ein Bit auf den Bus gegeben. Der Bus wird dadurch aktiv (dominant). Es fließt ein Strom durch die Abschlusswiderstände und erzeugt eine Differenzspannung zwischen den beiden Busleitungen. Die rezessiven und dominanten Zustände werden in den Busknoten in entsprechende Spannungen umgewandelt und von den Empfängerschaltkreisen erkannt.



Grafik: Buspegel

Durch diese differentielle Übertragung mit gemeinsamem Rückleiter wird die Übertragungssicherheit entscheidend verbessert. Störspannungen, die von außen auf das System einwirken, oder Massepotential-Verschiebungen beeinflussen beide Signalleitungen mit gleichen Störgrößen. Dadurch fallen die Störungen bei der Differenzbildung im Empfänger wieder heraus.

© ifm electronic gmbh

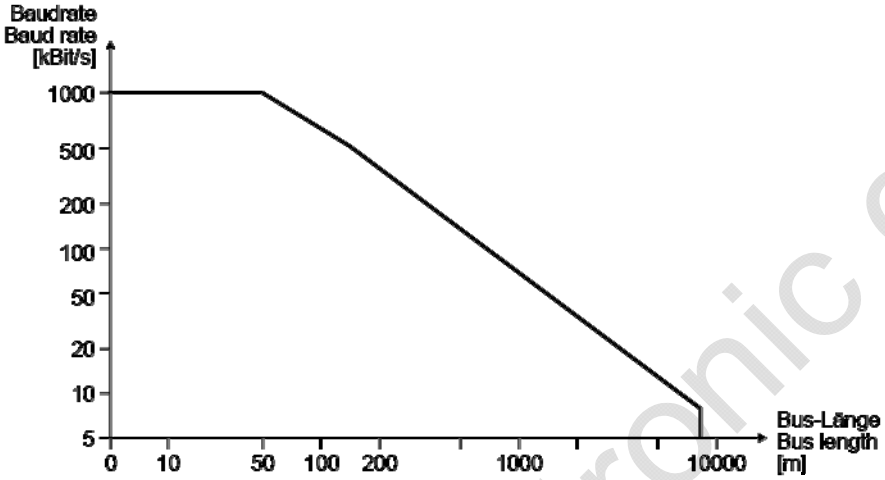
7.2.3 Busleitungslänge

1180

Die Länge der Busleitung ist abhängig von:

- Beschaffenheit der Busverbindung (Kabel, Steckverbinder),
- Leitungswiderstand,
- benötigte Übertragungsrate (Baud-Rate),
- Länge der Stichleitungen.

Vereinfachend kann man von folgender Abhängigkeit zwischen Buslänge und Baud-Rate ausgehen:



Grafik: Busleitungslänge

Baud-Rate [kBit/s]	Buslänge [m]	nominelle Bit-Länge [µs]
1 000	30	1
800	50	1,25
500	100	2
250	250	4
125	500	8
62,5	1 000	20
20	2 500	50
10	5 000	100

Tabelle: Abhängigkeiten Buslänge / Baudrate / Bitzeit

© ifm electronic gmbh

7.2.4 Leitungsquerschnitte

1181

Für die Auslegung des CAN-Netzes ist auch der Leitungsquerschnitt der eingesetzten Busleitung zu beachten. Die folgende Tabelle beschreibt die Abhängigkeit des Leiterquerschnitts bezogen auf die Leitungslänge und der Anzahl der daran angeschlossenen Teilnehmer (Knoten).

Leitungslänge [m]	**Leiterquerschnitt** [mm²] **bei 32 Knoten**	**Leiterquerschnitt** [mm²] **bei 64 Knoten**	**Leiterquerschnitt** [mm²] **bei 100 Knoten**
≤ 100	0,25	0,25	0,25
≤ 250	0,34	0,50	0,50
≤ 500	0,75	0,75	1,00

Abhängig von den EMV-Anforderungen können Sie die Busleitungen wie folgt ausführen:
- parallel,
- als Twisted-Pair
- und/oder abgeschirmt.

© ifm electronic gmbh

7.3 CAN-Datenaustausch

Inhalt:

Hinweise 93
Daten empfangen 95
Daten senden 95

1168

Der CAN-Datenaustausch erfolgt über das in der ISO 11898 international genormte CAN-Protokoll der Verbindungsschicht (Ebene 2) des siebenschichtigen ISO/OSI-Referenzmodells.

Jeder Bus-Teilnehmer kann Nachrichten senden (Multimaster-Fähigkeit). Der Datenaustausch arbeitet ähnlich dem Rundfunk. Daten werden ohne Absender und Adresse auf den Bus gesendet. Die Daten sind lediglich durch ihren Identifier gekennzeichnet. Es ist Aufgabe jedes Teilnehmers, die gesendeten Daten zu empfangen und an Hand des Identifiers zu prüfen, ob die Daten für diesen Teilnehmer relevant sind. Dieser Vorgang wird vom CAN-Controller in Verbindung mit dem Betriebssystem automatisch durchgeführt.

Für den normalen CAN-Datenaustausch muss der Programmierer lediglich bei der Softwareerstellung die Datenobjekte mit ihren Identifiern dem System bekannt machen. Dies erfolgt über folgende FBs:

- *CAN_x_RECEIVE* (→ Seite 97) (CAN-Daten empfangen) und
- *CAN_x_TRANSMIT* (→ Seite 99) (CAN-Daten senden).

Über diese FBs werden folgende Einheiten zu einem Datenobjekt verknüpft:

- die RAM-Adresse der Arbeitsdaten,
- der Datentyp,
- der gewählte Identifier (ID).

Diese Datenobjekte nehmen am Datenaustausch über den CAN-Bus teil. Die Sende- und Empfangsobjekte können aus allen gültigen IEC-Datentypen (z.B. BOOL, WORD, INT, ARRAY) definiert werden.

Die CAN-Nachricht besteht aus einem CAN-Identifier (*CAN-ID*, → Seite 93) und maximal 8 Datenbytes. Der ID repräsentiert nicht das Absender- oder Empfängermodul, sondern kennzeichnet die Nachricht. Um Daten zu übertragen, ist es notwendig, dass im Sendemodul ein Sendeobjekt und in mindestens einem anderen Modul ein Empfangs-Objekt deklariert ist. Beide Deklarationen müssen dem gleichen Identifier zugeordnet sein.

© ifm electronic gmbh

7.3.1 Hinweise

8394

CAN-ID

1166

Je nach CAN-ID sind folgende CAN-Identifier frei verfügbar für den Datentransfer:

CAN-ID base	CAN-ID extended
11 Bit	29 Bit
2 047 CAN-Identifier	536 870 912 CAN-Identifier
Standard-Applikationen	Motor-Management (SAE J1939), Truck & Trailer Interface (ISO 11992)

HINWEIS

Der 29-Bit-CAN-ID steht bei einigen Geräten nicht für alle CAN-Schnittstellen zur Verfügung,
→ Datenblatt.

Derselbe CAN-Controller kann NICHT gleichzeitig 11 Bit und 29 Bit lange CAN-Identifier empfangen.

Wir empfehlen: In einem CAN-Netzwerk ausschließlich 11 Bit lange CAN-Identifier benutzen ODER 29 Bit lange CAN-Identifier.

Beispiel 11-Bit CAN-ID (base):

SOF	CAN-ID base Bit 28 ... Bit 18										RTR	IDE
0	0	0	0	0	1	1	1	1	1	1	0	0
	0			7				F				

Beispiel 29-Bit CAN-ID (extended):

SOF	CAN-ID base Bit 28 ... Bit 18										SRR	IDE	CAN-ID extended Bit 17 ... Bit 0																RTR
0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0		1			F				C			0			0			0			0							

Legende:
SOF = Start of frame
Flanke von rezessiv zu dominant

RTR = **R**emote **t**ransmission **r**equest
dominant: Diese Nachricht liefert Daten
rezessiv: Diese Nachricht fordert Daten an

IDE = **Id**entifier **e**xtension flag
dominant: Hiernach folgen Steuerungs-Bits
rezessiv: Hiernach folgt der zweite Teil des 29-Bit-Identifier

SRR = **S**ubstitute **r**emote **r**equest
rezessiv: Extended CAN-ID: Ersetzt das RTR-Bit an dieser Stelle

© ifm electronic gmbh

Zusammenfassung CAN / CANopen

3956

- Der COB-ID der Netzwerkvariablen muss sich unterscheiden vom CANopen-Slave-ID in der Steuerungskonfiguration und von den IDs der FBs CANx_TRANSMIT und CANx_RECEIVE!
- Wenn mehr als 8 Bytes von Netzwerkvariablen in einen COB-ID gepackt werden, erweitert CANopen das Datenpaket automatisch auf mehrere aufeinander folgende COB-IDs. Dies kann zu Konflikten mit manuell definierten COB-IDs führen!
- Netzwerkvariable können keine String-Variablen transportieren.
- Netzwerkvariable können transportiert werden...
  - wenn eine Variable TRUE wird (Event),
  - bei Datenänderung in der Netzwerkvariablen oder
  - zyklisch nach Zeitablauf.
- Die Intervall-Zeit beschreibt die Periode zwischen Übertragungen bei zyklischer Übertragung. Der Mindestabstand beschreibt die Wartezeit zwischen zwei Übertragungen, wenn die Variable sich zu oft ändert.
- Um die Buslast zu mindern, die Nachrichten via Netzwerkvariablen oder CANx_TRANSMIT mit Hilfe von verschiedenen Events auf mehrere SPS-Zyklen verteilen.
- Jeder Aufruf von CANx_TRANSMIT oder CANx_RECEIVE erzeugt ein Nachrichtenpaket von 8 Bytes.
- In der Steuerungskonfiguration sollten die Werte für [Com Cycle Period] und [Sync. Window Length] gleich groß sein. Diese Werte müssen größer sein als die SPS-Zykluzeit.
- Wenn die [Com Cycle Period] für einen Slave eingestellt ist, sucht der Slave in genau dieser Zeit nach einem Sync-Objekt des Masters. Deshalb muss der Wert für [Com Cycle Period] größer sein als die [Master Synch Time].
- Wir empfehlen, Slaves als "optional startup" und das Netzwerk als "automatic startup" zu setzen. Dies reduziert unnötige Buslast und ermöglicht einem kurzzeitig verlorenen Slave, sich wieder in das Netzwerk zu integrieren.
- Weil wir keinen Inhibit-Timer haben, empfehlen wir, Analog-Eingänge auf "synchrone Übertragung" zu setzen, um Busüberlastung zu vermeiden.
- Binäre Eingänge, insbesondere die unregelmäßig schaltenden, sollten am besten auf "asynchrone Übertragung" mittels Event-Timer gesetzt werden.
- Beim Überwachen des Slave-Status beachten:
  - Nach dem Starten von Slaves dauert es etwas, bis die Slaves "operational" sind.
  - Beim Abschalten des Systems können Slaves wegen vorzeitigem Spannungsverlust eine scheinbare Status-Änderung anzeigen.

© ifm electronic gmbh

7.3.2 Daten empfangen

1169

Grundsätzlich werden die empfangenen Datenobjekte automatisch (also ohne Einfluss durch den Anwender) in einem Zwischenspeicher abgelegt.

Pro Identifier steht ein solcher Zwischenspeicher (Warteschlange) zur Verfügung. Dieser Zwischenspeicher wird in Abhängigkeit von der Anwendersoftware nach dem FiFo-Prinzip (**F**irst **I**n, **F**irst **O**ut) über *CAN_x_RECEIVE* (→ Seite 97) entleert.

7.3.3 Daten senden

1170

Durch den Aufruf von *CAN_x_TRANSMIT* (→ Seite 99) übergibt das Applikations-Programm genau eine CAN-Nachricht an den CAN-Controller. Als Rückgabe erhält man die Information, ob die Nachricht erfolgreich an den CAN-Controller übergeben wurde. Dieser führt dann selbständig die eigentliche Übergabe der Daten auf den CAN-Bus aus.

Der Sendeauftrag wird abgewiesen, wenn der Controller nicht bereit ist, weil er bereits ein Datenobjekt überträgt. Der Sendeauftrag muss dann durch das Applikations-Programm wiederholt werden. Der Anwender bekommt diese Information durch ein Bit angezeigt.

Bei mehreren zeitgleich zum Senden bereiten CAN-Nachrichten wird die Nachricht mit dem niedrigsten ID vorrangig gesendet. Der Programmierer muss daher den *CAN-ID* (→ Seite 93) sehr umsichtig vergeben.

© ifm electronic gmbh

7.4 Beschreibung der CAN-Standardbausteine

Inhalt:

CAN_x_RECEIVE 97
CAN_x_TRANSMIT 99

3699

Hier werden die CAN-Funktionsblöcke zur Nutzung im Applikations-Programm beschrieben.

HINWEIS

Um die volle Leistungsfähigkeit von CAN zu nutzen, ist es unbedingt erforderlich, dass sich der Programmierer vor Beginn seiner Arbeit ein genaues **Buskonzept** aufbaut:

- Wie viele Datenobjekte mit welchen Identifiern werden benötigt?
- Wie soll das Gerät auf mögliche CAN-Fehler reagieren?
- Wie oft müssen Daten übertragen werden? Dem entsprechend oft müssen *CAN_x_TRANSMIT* (→ Seite 99) und *CAN_x_RECEIVE* (→ Seite 97) aufgerufen werden.

► Dabei überwachen, ob die Sendeaufträge erfolgreich an CAN_x_TRANSMIT übergeben wurden (Ausgang RESULT) oder dafür sorgen, dass die empfangenen Daten mit CAN_x_RECEIVE aus dem Datenpuffer der Warteschlange ausgelesen und sofort im übrigen Programm entsprechend verarbeitet werden.

Damit eine Kommunikationsverbindung aufgebaut werden kann, muss zuvor bei allen Teilnehmern des CAN-Netzwerkes die gleiche Übertragungsrate (Baud-Rate) eingestellt werden. Beim PDM360 wird diese in der Steuerungskonfiguration (→ *Steuerungskonfiguration aktivieren (z.B. CR1051)*, → Seite 41) vorgenommen.

Unabhängig davon, ob die Geräte eine oder mehrere CAN-Schnittstellen unterstützen, werden die der Schnittstelle zugehörigen Funktionen durch Nummerierung in der CAN-Funktion gekennzeichnet (z.B. CAN_1_TRANSMIT oder CAN_2_RECEIVE). In der Dokumentation wird aus Vereinfachungsgründen die Bezeichnung (z.B. CAN_x_TRANSMIT) für alle Varianten verwendet.

Beim Installieren der **ecomatmobile**-DVD "Software, tools and documentation" wurden auch Projekte mit Vorlagen auf Ihrem Computer im Programmverzeichnis abgelegt:
`…\ifm electronic\CoDeSys V…\Projects\Template_CDV…`

► Die gewünschte dort gespeicherte Vorlage in CoDeSys öffnen mit:
[Datei] > [Neu aus Vorlage…]

> CoDeSys legt ein neues Projekt an, dem der prinzipielle Programmaufbau entnommen werden kann. Es wird dringend empfohlen, dem gezeigten Schema zu folgen.
→ Kapitel *Programmiersystem über Templates einrichten*, → Seite 44

In diesem Beispiel werden über die Identifier 1 und 2 Datenobjekte mit einem weiteren CAN-Teilnehmer ausgetauscht. Dazu muss im anderen Teilnehmer zum Sende-Identifier ein Empfangs-Identifier (oder umgekehrt) existieren.

© ifm electronic gmbh

7.4.1 CAN_x_RECEIVE

3704

Baustein-Typ = Funktionsblock (FB)

x = Nr. 1...n der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Symbol in CoDeSys:

CAN_1_RECEIVE

9332

Enthalten in Bibliothek: `ifm_CAN1_LAYER2_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360smart: CR1070, CR1071

Beschreibung

3707

CAN_x_RECEIVE konfiguriert ein Datenempfangsobjekt und liest den Empfangspuffer des Datenobjektes aus.

Der FB muss für jedes Datenobjekt in der Initialisierungsphase einmalig aufgerufen werden, um dem CAN-Controller die Identifier der Datenobjekte bekannt zu machen.

Im weiteren Programmzyklus wird CAN_x_RECEIVE zum Auslesen des jeweiligen Empfangspuffers aufgerufen, bei langen Programmzyklen auch mehrfach. Der Programmierer muss durch Auswertung des Bytes AVAILABLE dafür Sorge tragen, dass neu eingegangene Datenobjekte aus dem Puffer abgerufen und weiterverarbeitet werden.

Jeder Aufruf des FB dekrementiert das Byte AVAILABLE um 1. Ist der Wert von AVAILABLE gleich 0, sind keine Daten im Puffer.

Durch Auswerten des Ausgangs OVERFLOW kann ein Überlauf des Datenpuffers erkannt werden. Wenn OVERFLOW = TRUE, dann ist mindestens 1 Datenobjekt verloren gegangen.

HINWEIS

Gilt nur für folgende Geräte:
- PDM360compact: CR1053, CR1056
- PDM360smart: CR1070, CR1071:

Um die CAN-Layer 2-Funktionen nutzen zu können, muss immer ein CANopen-Master in der Steuerungskonfiguration angelegt werden.

© ifm electronic gmbh

Parameter der Eingänge

631

Parameter	Datentyp	Beschreibung
CONFIG	BOOL	TRUE (nur 1 Zyklus lang): Datenobjekt konfigurieren FALSE: diese Funktion wird nicht ausgeführt
CLEAR	BOOL	TRUE: löscht den Datenpuffer (Warteschlange) FALSE: diese Funktion wird nicht ausgeführt
ID	WORD	Nummer des Datenobjekt-Identifier Zulässige Werte: 0...2 047

Parameter der Ausgänge

3712

Parameter	Datentyp	Beschreibung
DATA	ARRAY[0...7] OF BYTE	Das Array enthält maximal 8 Datenbytes
DLC	BYTE	Anzahl der übertragenen Bytes im Array DATA Mögliche Werte: 0...8.
RTR	BYTE	Nicht bei PDM360smart: CR1070, CR1071 Rückmeldung Remote Frame
AVAILABLE	BYTE	Anzahl der eingegangenen Meldungen
OVERFLOW	BOOL	TRUE: Überlauf des Datenpuffers ⇨ Datenverlust! FALSE: Puffer noch nicht gefüllt

© ifm electronic gmbh

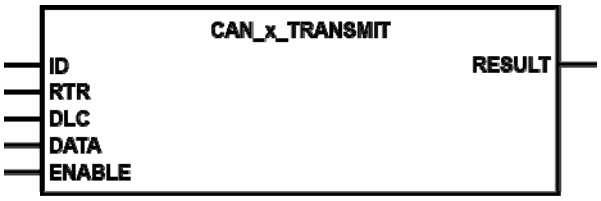
7.4.2 CAN_x_TRANSMIT

3701

Baustein-Typ = Funktionsblock (FB)

x = Nr. 1...n der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Symbol in CoDeSys:



CAN_1_TRANSMIT

9334

Enthalten in Bibliothek: `ifm_CAN1_LAYER2_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360smart: CR1070, CR1071

Beschreibung

3706

CAN_x_TRANSMIT übergibt ein CAN-Datenobjekt (Message) an den CAN-Controller zur Übertragung.

Der FB wird für jedes Datenobjekt im Programmzyklus aufgerufen, bei langen Programmzyklen auch mehrfach. Der Programmierer muss durch Auswertung des FB-Ausgangs RESULT dafür Sorge tragen, dass sein Sendeauftrag auch angenommen wurde. Vereinfacht gilt bei 125 kBit/s, dass pro 1 ms ein Sendeauftrag ausgeführt werden kann.

Über den Eingang ENABLE kann die Ausführung des FB zeitweilig gesperrt werden (ENABLE = FALSE). Damit kann z.B. eine Busüberlastung verhindert werden.

Mehrere Datenobjekte können quasi gleichzeitig verschickt werden, wenn jedem Datenobjekt ein Flag zugeordnet wird und mit diesem die Ausführung des FB über den ENABLE-Eingang gesteuert wird.

HINWEIS

Gilt nur für folgende Geräte:
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360smart: CR1071:

Um die CAN-Layer 2-Funktionen nutzen zu können, muss immer ein CANopen-Master in der Steuerungskonfiguration angelegt werden.

Die Nutzung der 2. CAN-Schnittstelle muss durch Eintrag eines 2. Masters erfolgen.

© ifm electronic gmbh

Parameter der Eingänge

3708

Parameter	Datentyp	Beschreibung
ID	WORD	Nummer des Datenobjekt-Identifier Zulässige Werte: 0...2 047
RTR	BOOL	Nicht bei PDM360smart: Remote Frame: Besondere Anforderung vom Master an bestimmten Slave zwecks z.B. Nodeguarding Die Eingänge DLC und DATA bleiben hierfür unbenutzt.
DLC	BYTE	Anzahl der zu übertragenden Bytes aus dem Array DATA Zulässige Werte: 0...8
DATA	ARRAY[0...7] OF BYTE	Das Array enthält maximal 8 Datenbytes
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv

Parameter der Ausgänge

3709

Parameter	Datentyp	Beschreibung
RESULT	BOOL	TRUE: Sendeauftrag ist angenommen FALSE: Sendeauftrag ist abgelehnt Grund: z.B. Puffer ist voll

© ifm electronic gmbh

7.5 ifm-CANopen-Bibliotheken

Inhalt:

Technisches zu CANopen 101
Bibliotheken für CANopen 141

1856

HINWEIS

Folgende Geräte unterstützen CANopen nur für die 1. CAN-Schnittstelle:
- CabinetController: CR0303
- ClassicController: CR0020, CR0505
- ExtendedController: CR0200 (1. und 3. CAN-Schnittstelle)
- SafetyController: CR7021, CR7506
- ExtendedSafetyController: CR7201 (1. und 3. CAN-Schnittstelle)
- SmartController: CR25nn

Wurde bereits der CANopen-Master eingefügt, kann das Gerät nicht mehr als CANopen-Slave über CoDeSys genutzt werden.

Die Implementierung eines eigenen Protokolls auf Schnittstelle 2 oder Nutzung des Protokolls nach SAE J1939 oder ISO 11992 ist aber jederzeit möglich, sofern auf dem Gerät verfügbar (→ Kapitel *Verfügbare CAN-Schnittstellen und CAN-Protokolle*, → Seite 85).

7.5.1 Technisches zu CANopen

Inhalt:

CANopen Netzwerk-Konfiguration, Status- und Fehlerbehandlung 102
CANopen-Unterstützung durch CoDeSys 103
CANopen-Master 105
CANopen-Slave 126
CANopen-Netzwerkvariablen 135

7773

CANopen-Tabellen (→ Seite 290) zur Übersicht finden Sie im Anhang.

© ifm electronic gmbh

CANopen Netzwerk-Konfiguration, Status- und Fehlerbehandlung

7471

Bei allen programmierbaren Geräten wird die CANopen-Schnittstelle von CoDeSys eingesetzt. Während Sie die Netzwerkkonfiguration und die Parametrierung der angeschlossenen Geräte direkt über die Programmiersoftware vornehmen, können die Fehlermeldungen nur über verschachtelte Variablenstrukturen im CANopen-Stack erreicht werden. Die nachfolgende Dokumentation zeigt Ihnen den Aufbau und die Anwendung der Netzwerkkonfiguration und beschreibt die Bausteine der ifm CANopen-Gerätebibliotheken.

Die Kapitel *CANopen-Unterstützung durch CoDeSys* (→ Seite 103), *CANopen-Master* (→ Seite 105), *CANopen-Slave* (→ Seite 126) und *CANopen-Netzwerkvariablen* (→ Seite 135) beschreiben die internen Bausteine des CoDeSys-CANopen-Stacks und ihre Anwendung. Außerdem bekommen Sie einen Einblick über die Anwendung des Netzwerkkonfigurators.

Die Kapitel über die Bibliotheken `ifm_CRnnnn_CANopenMaster_Vxxyyzz.lib` und `ifm_CRnnnn_CANopenSlave_Vxxyyzz.lib` beschreiben alle Bausteine zur Fehlerverarbeitung und zur Abfrage des Gerätestatus beim Einsatz als Master oder Slave.

HINWEIS

Unabhängig vom eingesetzten Gerät haben alle Bibliotheken den gleichen Aufbau der Funktionsschnittstellen. Die geringfügigen Unterschiede (z.B. CANOPEN_LED_STATUS) werden direkt in den jeweiligen Bausteinen beschrieben.

Es ist zwingend notwendig, dass Sie nur die jeweilige gerätespezifische Bibliothek einsetzen. Den Zusammenhang können Sie an der integrierten Geräte-Artikelnummer erkennen.
Beispiel CR0020: → `ifm_CR0020_CANopenMaster_Vxxyyzz.lib`

→ Kapitel *Target einrichten*, → Seite 39

Bei Verwendung anderer Bibliotheken kann das Gerät nicht mehr richtig funktionieren.

© ifm electronic gmbh

CANopen-Unterstützung durch CoDeSys

1857

Allgemeines zu CANopen mit CoDeSys

2075

CoDeSys ist eines der führenden Systeme für die Programmierung von Steuerungssystemen nach dem internationalem Standard IEC 61131. Um CoDeSys für den Anwender interessanter zu gestalten, wurden viele wichtige Funktionen in das Programmiersystem integriert, darunter auch ein Konfigurator für CANopen. Mit diesem CANopen-Konfigurator können Sie CANopen-Netzwerke (in einigen Punkten eingeschränkt) unter CoDeSys konfigurieren.

CANopen ist als CoDeSys-Bibliothek in IEC 61131-3 implementiert. Die Bibliothek stützt sich auf sehr einfache Basis-CAN-Funktionen ab, die als CAN-Treiber bezeichnet werden.

Durch die Realisierung der CANopen-Funktionen als CoDeSys-Bibliothek ist eine einfache Skalierung des Zielsystems möglich. So verbraucht die CANopen-Funktion nur dann Zielsystem-Ressourcen, wenn die Funktion auch wirklich genutzt wird. Zur weiteren Schonung von Zielsystem-Ressourcen wird durch CoDeSys automatisch eine genau der Konfiguration entsprechende Datenbasis für die CANopen-Master-Funktion generiert.

Ab der Programmiersystemversion CoDeSys Version 2.3.6.0 kann ein **ecomat*mobile***-Controller als CANopen-Master und als CANopen-Slave genutzt werden.

HINWEIS

Für alle **ecomat*mobile***-Controller und das PDM360smart müssen Sie die CANopen-Bibliotheken mit folgendem Zusatz einsetzen:

- Für CR0032 Target-Version bis V01, alle anderen Geräte bis V04.00.05: "**OptTable**"
- Für CR0032 Target-Version ab V02, alle anderen Geräte ab V05: "**OptTableEx**"

Wenn Sie ein Projekt neu anlegen, werden diese Bibliotheken im Allgemeinen automatisch geladen. Sollten Sie selbst die Bibliotheken über die Bibliotheksverwaltung einfügen, müssen Sie auf die korrekte Auswahl achten.

Die CANopen-Bibliotheken ohne diesen Zusatz werden für alle anderen programmierbaren Geräte genutzt (z.B. PDM360compact).

© ifm electronic gmbh

CANopen Begriffe und Implementation

1858

Nach der CANopen-Spezifikation gibt es keine Master und Slaves in einem CAN-Netz. Stattdessen gibt es nach CANopen einen NMT-Master (NMT = Netzwerk-Management), einen Konfigurationsmaster usw., immer mit der Vorstellung, dass alle Teilnehmer eines CAN-Netzes gleichberechtigt sind.

Die Implementierung geht davon aus, dass ein CAN-Netz als Peripherie einer CoDeSys-programmierbaren Steuerung dient. Demzufolge wird eine **ecomat*mobile***-Steuerung oder ein PDM360-Display im CAN-Konfigurator von CoDeSys als CANopen-Master bezeichnet. Dieser Master ist NMT-Master und Konfigurationsmaster. Im Normalfall wird der Master dafür sorgen, dass das Netz in Betrieb genommen werden kann. Er übernimmt die Initiative, die einzelnen Nodes (= Netzwerk-Knoten) zu starten, die ihm per Konfiguration bekannt sind. Diese Nodes werden als Slaves bezeichnet.

Um den Master ebenfalls dem Status eines CANopen-Slaves näherzubringen, wurde ein Objektverzeichnis für den Master eingeführt. Auch kann der Master als SDO-Server (SDO = Service Data Object) auftreten und nicht nur in der Konfigurationsphase der Slaves als SDO-Client.

IDs (Adressen) in CANopen

3952

In CANopen werden diverse Arten von 'Adressen' (hier: IDs) unterschieden:

- COB-ID
  Der **C**ommunication-**Ob**ject-**Id**entifier adressiert die Nachricht (= das Kommunikationsobjekt) im Geräteverzeichnis. Ein Kommunikationsobjekt besteht aus einem oder mehreren CAN-Nachrichten mit bestimmten Aufgaben, z.B.:
  - PDO (**P**rocess **D**ata **O**bject = Nachrichten-Objekt mit Prozessdaten),
  - SDO (**S**ervice **D**ata **O**bject = Nachrichten-Objekt mit Servicedaten),
  - Emergency (Nachrichten-Objekt mit Notfalldaten),
  - Time (Nachrichten-Objekt mit Zeitangaben) oder
  - Error Control (Nachrichten-Objekt mit Fehlermeldungen).
- CAN-ID
  Der **CAN-Id**entifier definiert netzwerkweit CAN-Nachrichten. Der CAN-ID ist Hauptbestandteil des Arbitration-Feldes eines CAN-Datenübertragungsblocks. Je niedriger der CAN-ID, desto höher die Priorität der Meldung.
- Download-ID
  Der Download-ID bezeichnet den Node-ID für Service-Kommunikation per SDO für den Programm-Download und das Debuggen.
- Node-ID
  Der **Node-Id**entifier ist ein eindeutiger Bezeichner für CANopen-Geräte (Devices) im CAN-Netzwerk. Der Node-ID ist auch Bestandteil einiger vordefinierter Verbindungssätze (→ *Funktions-Code / Predefined Connectionset*, → Seite 293).

Vergleich Download-ID vs. COB-ID:

Controller Programm-Download		CANopen	
Download-ID	**COB-ID SDO**	**Node-ID**	**COB-ID SDO**
1…127	TX: 580_{16} + Download-ID	1…127	TX: 580_{16} + Node-ID
	RX: 600_{16} + Download-ID		RX: 600_{16} + Node-ID

TX = Slave sendet an Master
RX = Slave empfängt von Master

©ifm electronic gmbh

CANopen-Master

Inhalt:

Abgrenzung zu anderen CANopen-Bibliotheken 105
Ein CANopen-Projekt erstellen 107
CANopen-Slaves einfügen und konfigurieren 111
Der Master zur Laufzeit 115
Netzwerk starten 117
Netzwerkzustände 118

1859

Abgrenzung zu anderen CANopen-Bibliotheken

1990

Die von 3S (Smart Software Solutions) realisierte CANopen-Bibliothek grenzt sich in verschiedenen Punkten von auf dem Markt befindlichen Systemen ab. Sie wurde nicht entwickelt, um andere Bibliotheken namhafter Hersteller überflüssig zu machen, sondern ist bewusst für den Einsatz mit dem CoDeSys-Programmier- und Laufzeitsystem optimiert.

Die Bibliotheken wurden nach der Spezifikation der CiA DS301, V402 erstellt.

Für Sie als Anwender der CoDeSys-CANopen-Bibliothek ergeben sich folgende Vorteile:

- Die Implementierung ist unabhängig vom Zielsystem und damit praktisch auf jeder mit CoDeSys-programmierbaren Steuerung direkt verwendbar.
- Das komplette System beinhaltet den CANopen-Konfigurator und die Einbindung in das Entwicklungssystem.
- Die CANopen-Funktionalität ist nachladbar. Das bedeutet, dass die CANopen-Funktionen ohne Änderung des Betriebssystems geladen und aktualisiert werden können.
- Die Ressourcen des Zielsystems werden geschont, da nicht die Ressourcen für eine Maximalkonfiguration vorgehalten werden.
- Automatisches Aktualisieren der Ein- und Ausgänge ohne zusätzliche Maßnahmen.

Folgende in CANopen definierten Funktionen werden zurzeit von der **ifm**-CANopen-Bibliothek unterstützt:

- **PDOs Senden:** Master sendet zu den Slaves (Slave = Knoten, Device)
  Senden ereignisgesteuert (d.h. bei Änderung), zeitgesteuert (RepeatTimer) oder als synchrone PDOs, d.h. immer wenn ein SYNC vom Master gesendet wurde. Auch eine externe SYNC-Quelle kann benutzt werden, um das Senden von synchronen PDOs zu initiieren.
- **PDOs Empfangen:** Master empfängt vom Slave
  Je nach Slave: ereignisgesteuert, abfragegesteuert, azyklisch und zyklisch.
- **PDO-Mapping**
  Zuordnung zwischen lokalem Objektverzeichnis und PDOs vom/zum CANopen-Slave (wenn vom Slave unterstützt).
- **SDO Senden und Empfangen** (unsegmentiert, d.h. 4 Bytes pro Objektverzeichnis-Eintrag)
  Automatische Konfiguration aller Slaves über SDOs beim Systemstart.
  Applikationsgesteuertes Senden und Empfangen von SDOs zu konfigurierten Slaves.
- **Synchronisation**
  Automatisches Senden von SYNC-Nachrichten durch den CANopen-Master.

© ifm electronic gmbh

- **Nodeguarding**
  Automatisches Senden von Guarding-Nachrichten und Überwachung der Lifetime für jeden entsprechend konfigurierten Slave.
  Wir empfehlen: Für aktuelle Geräte besser mit Heartbeat arbeiten, weil dann die Buslast niedriger ist.
- **Heartbeat**
  Automatisches Senden und Überwachen von Heartbeat-Nachrichten.
- **Emergency**
  Empfangen und Speichern von Emergency-Nachrichten von den konfigurierten Slaves.
- **Node-ID** und **Baudrate** in den Slaves setzen
  Durch Aufruf einer einfachen Funktion können Node-ID und Baudrate eines Slaves zur Laufzeit der Applikation gesetzt werden.

Folgende in CANopen definierten Funktionen werden von der 3S (Smart Software Solutions) CANopen-Bibliothek derzeit **nicht** unterstützt:

- Dynamische Identifier-Zuordnung
- Dynamische SDO-Verbindungen
- Blockweiser SDO-Transfer, segmentierter SDO-Transfer (die Funktionalität kann mit *CANx_SDO_READ* (→ Seite 164) und *CANx_SDO_WRITE* (→ Seite 166) in der jeweiligen **ifm**-Gerätebibliothek realisiert werden).
- Alle oben nicht genannten Möglichkeiten des CANopen Protokolls.

© ifm electronic gmbh

Ein CANopen-Projekt erstellen

1860

Die Erstellung eines neuen Projektes mit einem CANopen-Master wird nachfolgend schrittweise beschrieben. Dabei gehen wir davon aus, dass Sie CoDeSys auf dem Rechner bereits fertig installiert haben und die Target- und EDS-Dateien ebenfalls richtig installiert oder kopiert wurden.

Eine weitergehende detaillierte Beschreibung zur Einstellung und Anwendung des Dialogs Steuerungs- und CANopen-Konfiguration → CoDeSys-Handbuch unter [Ressourcen] > [Steuerungskonfiguration] und in der Online-Hilfe.

► Nach der Neuanlage eines Projektes (→ Kapitel *Target einrichten*, → Seite 39) in der Steuerungskonfiguration über [Einfügen] > [Unterelement anhängen] den CANopen-Master einfügen.

> Bei Steuerungen mit 2 oder mehr CAN-Schnittstellen wird automatisch Schnittstelle 1 für den Master konfiguriert.

> Die folgenden Bibliotheken und Software-Module werden automatisch eingebunden:

- die `STANDARD.LIB`, welche die in der IEC-61131 definierten Standardfunktionen für die Steuerung zu Verfügung stellt,
- die `3S_CanOpenManager.LIB`, welche die CANopen-Basisfunktionalitäten zur Verfügung stellt
  (ggf. `3S_CanOpenManagerOptTable.LIB` für C167-Controller),
- eine oder mehrere der Bibliotheken `3S_CANopenNetVar.LIB`, `3S_CANopenDevice.LIB` und `3S_CANopenMaster.LIB` (ggf. `3S_...OptTable.LIB` für C167-Controller), je nach gewünschter Funktionalität,
- die Systembibliotheken `SysLibSem.LIB` und `SysLibCallback.LIB`.

► Um die vorbereiteten Netzwerkdiagnose-, Status- und EMCY-Funktion zu nutzen, die Bibliothek `ifm_CRnnnn_CANopenMaster_Vxxyyzz.LIB` manuell im Bibliotheksverwalter einfügen. Ohne diese Bibliothek müssen Sie die Netzwerkinformationen direkt aus den verschachtelten Strukturen der CoDeSys-CANopen-Bibliotheken auslesen.

► Zusätzlich die folgenden Bibliotheken und Software-Module einbinden:

- die Gerätebibliothek für die jeweilige Hardware, z.B. `ifm_CR0020_Vxxyyzz.LIB`. Diese Bibliothek stellt alle gerätespezifischen Funktionen zur Verfügung.
- EDS-Dateien für alle Slaves, die am Netzwerk betrieben werden sollen. Die EDS-Dateien für alle **ifm**-CANopen-Slaves stellt die **ifm electronic gmbh** zur Verfügung (→ Kapitel *Programmiersystem über Templates einrichten*, → Seite 44).
  Für die EDS-Dateien von Fremd-Knoten ist der jeweilige Hersteller verantwortlich.

© ifm electronic gmbh

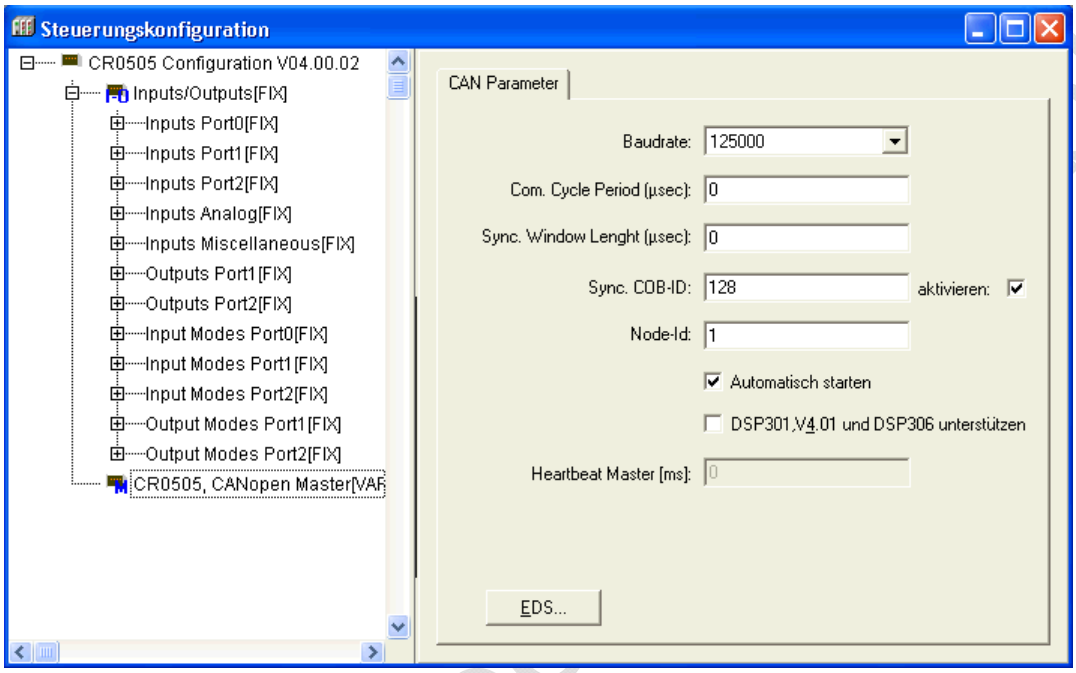
CANopen-Master: Register [CAN-Parameter]

1967

In diesem Dialogfenster können für den Master die wichtigsten Parameter eingestellt werden. Bei Bedarf kann über die Schaltfläche [EDS...] der Inhalt der Master-EDS-Datei angesehen werden.

Diese Schaltfläche wird nur angezeigt, wenn die EDS-Datei (z.B. `CR0020MasterODEntry.EDS`) im Verzeichnis `...\CoDeSys V2.3\Library\PLCConf` vorhanden ist.

Aus dieser EDS-Datei wird bei der Übersetzung des Applikations-Programms automatisch das Objektverzeichnis des Masters erzeugt.



Beispiel: Steuerungskonfiguration für CR0505 CANopen-Master

CAN Parameter: Baudrate

10028

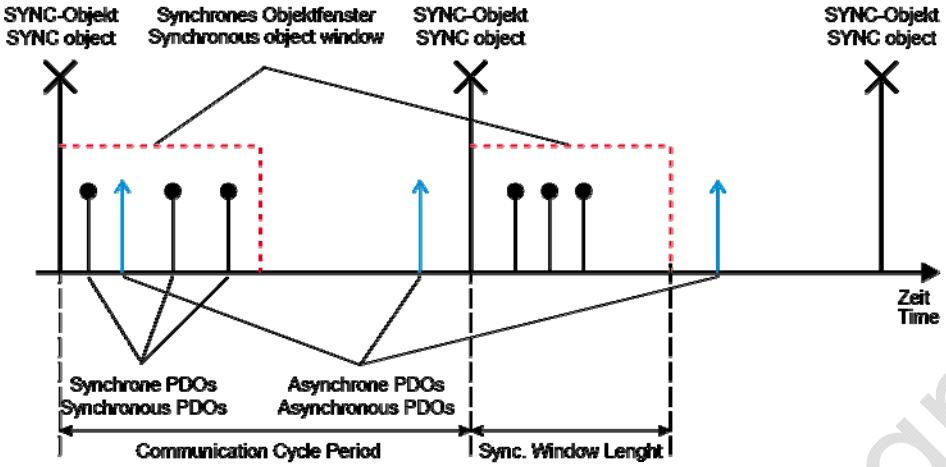
Wählen Sie an dieser Stelle bitte die Baudrate für den Master aus.
Die Baudrate muss der Übertragungsgeschwindigkeit der anderen Netzwerkteilnehmer entsprechen.

© ifm electronic

CAN Parameter: Communication Cycle Period / Sync. Window Length

10029

Nach Ablauf der [Communication Cycle Period] wird eine SYNC-Nachricht vom Master verschickt:



Die [Sync. Window Length] gibt die Zeit an, in der synchrone PDOs von den anderen Netzwerkteilnehmern verschickt und vom Master empfangen werden müssen.

Da in den meisten Applikationen keine besonderen Anforderungen an das SYNC-Objekt gestellt werden, können Sie für die [Communication Cycle Period] und die [Sync. Window Length] die gleiche Zeit einstellen.
Bitte beachten Sie, dass die Zeit in [µsec] eingegeben wird (der Wert 50000 entspricht 50 ms).

CAN Parameter: Sync. COB-ID

10030

In diesem Feld kann der Identifier für die SYNC-Nachricht einstellt werden. Diese wird immer nach Ablauf der Communication Cycle Period verschickt. Der Defaultwert ist 128 und sollte im Normalfall nicht geändert werden. Um das Versenden der SYNC-Nachricht zu aktivieren, muss das Kontrollfeld [aktivieren] gesetzt sein.

HINWEIS

Die SYNC-Nachricht wird immer am Anfang eines Programmzyklus erzeugt. Danach werden die Eingänge gelesen, das Programm abgearbeitet, die Ausgänge geschrieben und zuletzt alle synchronen PDOs gesendet.

Bitte beachten Sie, dass sich die SYNC-Zeit verlängert, wenn die eingestellte SYNC-Zeit kürzer als die Programmzykluszeit ist.

Beispiel: Communication Cycle Period = 10 ms und Programmzykluszeit = 30 ms.
Die SYNC-Nachricht wird erst nach 30 ms versendet.

© ifm electronic gmbh

CAN Parameter: Node-ID

10031

Setzen Sie in diesem Feld die Knotennummer (nicht den Download-ID!) des Masters ein.
Die Knotennummer darf im Netzwerk nur einmal vorkommen, andernfalls kommt es zu Kommunikationsstörungen.

CAN Parameter: Automatisch starten

10032

Das Netzwerk und die angeschlossenen Knoten werden nach einer erfolgreichen Konfiguration in den Zustand "operational" gesetzt und damit gestartet.
Ist das Optionsfeld nicht angewählt, muss das Netzwerk manuell gestartet werden.

CAN Parameter: Heartbeat

10033

Wenn die anderen Teilnehmer im Netzwerk Heartbeat unterstützen, kann die Option [DSP301, V4.01... unterstützen] selektiert werden. Bei Bedarf kann der Master noch ein eigenes Heartbeat-Signal nach Ablauf der eingestellten Zeit erzeugen.

© ifm electronic gmbh

CANopen-Slaves einfügen und konfigurieren

Inhalt:

CANopen-Slave: Register [CAN Parameter] 112
Register [PDO-Mapping empfangen] und [PDO-Mapping senden] 113
Register [Service Data Objects] 114

1861

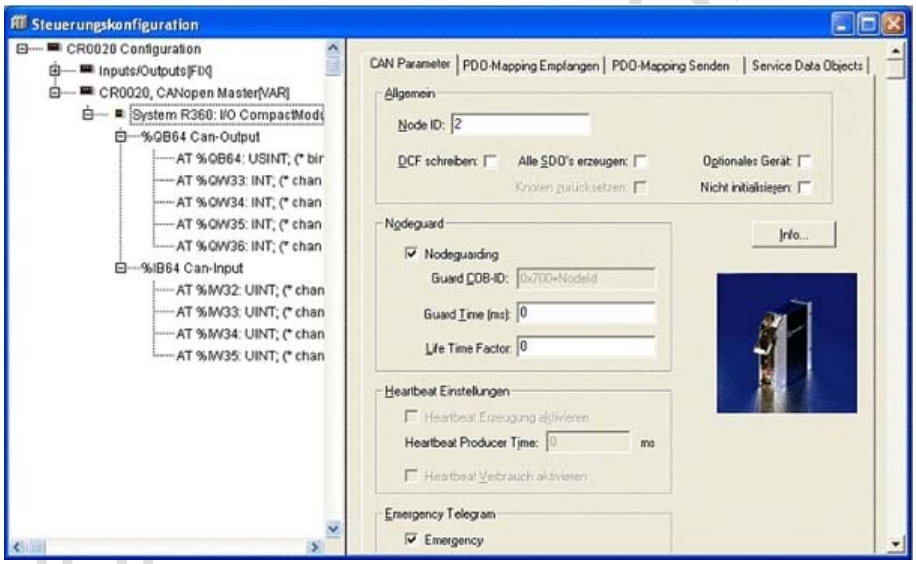
Als nächstes können Sie nun die CANopen-Slaves einfügen. Dazu müssen Sie erneut den Dialog in der Steuerungskonfiguration [Einfügen] > [Unterelement anhängen] aufrufen. Es steht Ihnen eine Liste der im Verzeichnis PLC_CONF gespeicherten CANopen-Gerätebeschreibungen (EDS-Dateien) zur Verfügung. Durch Auswahl des entsprechenden Gerätes wird dieses direkt in den Baum der Steuerungskonfiguration eingefügt.

HINWEIS

Wird ein Slave über den Konfigurationsdialog in CoDeSys hinzugefügt, wird für jeden Knoten dynamisch Quellcode in das Applikations-Programm integriert. Gleichzeitig verlängert jeder zusätzlich hinzugefügte Slave die Zykluszeit des Applikations-Programms. Das bedeutet: in einem Netzwerk mit vielen Slaves kann der Master keine weiteren zeitkritischen Aufgaben (z.B. den FB OCC_TASK) abarbeiten.

Ein Netzwerk mit 27 Slaves hat eine Grund-Zykluszeit von 30 ms.

Bitte beachten Sie, dass die maximale Zeit für einen SPS-Zyklus von ca. 50 ms nicht überschritten werden sollte (Watchdog-Zeit: 100 ms).



Beispiel: Steuerungskonfiguration für CR0020 CANopen-Master mit angeschlossenem I/O CompactModul

CANopen-Slave: Register [CAN Parameter]

1968

CAN Parameter: Node-ID

10036

Der Node-ID dient zur eindeutigen Identifizierung des CAN-Moduls und entspricht der am Modul eingestellten Nummer zwischen 1 und 127.

Der ID wird dezimal eingegeben und wird automatisch um eins erhöht, wenn Sie ein neues Modul hinzufügen.

CAN Parameter: DCF schreiben

10037

Ist [DCF schreiben] aktiviert, wird nach dem Einfügen einer EDS-Datei im eingestellten Verzeichnis für Übersetzungsdateien eine DCF-Datei erstellt, deren Namen sich zusammensetzt aus dem Namen der EDS-Datei und dem angehängten Node-ID.

CAN Parameter: Alle SDOs erzeugen

10038

Ist diese Option aktiviert, werden für alle Kommunikationsobjekte SDOs erzeugt.
Default-Werte werden nicht erneut geschrieben!

CAN Parameter: Knoten zurücksetzen

10039

Der Slave wird zurückgesetzt ("load"), sobald die Konfiguration in die Steuerung geladen wird.

CAN Parameter: Optionales Gerät

10040

Ist die Option [Optionales Gerät] aktiviert, versucht der Master nur einmal, von diesem Knoten zu lesen. Bei fehlender Antwort wird der Knoten ignoriert und der Master geht in den normalen Betriebszustand über.

Wird der Slave zu einem späteren Zeitpunkt an das Netzwerk angeschlossen und erkannt, wird er automatisch gestartet. Dazu müssen Sie die Option [Automatisch starten] in den CAN-Parametern des Masters angewählt haben.

CAN Parameter: Nicht initialisieren

10041

Wird diese Option aktiviert, nimmt der Master den Knoten sofort in Betrieb, ohne ihm Konfigurations-SDOs zu schicken. (Die SDO-Daten werden aber dennoch erzeugt und auf der Steuerung gespeichert.)

© ifm electronic gmbh

CAN Parameter: Nodeguarding- / Heartbeat-Einstellungen

10042

Je nach Gerät haben Sie die Wahl:
- [Nodeguarding] und [Life Time Factor] einstellen ODER
- [Heartbeat] einstellen.

Wir empfehlen: Für aktuelle Geräte besser mit Heartbeat arbeiten, weil dann die Buslast niedriger ist.

CAN Parameter: Emergency Telegram

10043

Die Option ist im Normalfall angewählt. Die EMCY-Nachrichten werden mit dem angegebenen Identifier übertragen.

CAN Parameter: Communication Cycle

10044

In ganz speziellen Anwendungsfällen können Sie an dieser Stelle eine Überwachungszeit für die vom Master erzeugten SYNC-Nachrichten einstellen.

Bitte beachten Sie, dass diese Zeit länger als die SYNC-Zeit des Masters sein muss. Der optimale Wert muss ggf. experimentell ermittelt werden.

Nodeguarding und Heartbeat reichen in den meisten Fällen zur Knotenüberwachung aus.

Register [PDO-Mapping empfangen] und [PDO-Mapping senden]

1969

Die Registerkarten [PDO-Mapping empfangen] und [PDO-Mapping senden] im Konfigurationsdialog eines CAN-Moduls ermöglichen es, dass in der EDS-Datei beschriebene "Mapping" (Zuordnung zwischen lokalem Objektverzeichnis und PDOs vom/zum CANopen-Slave) des Moduls zu verändern (wenn es vom CAN-Modul unterstützt wird).

Auf der linken Seite stehen alle "mapbaren" Objekte der EDS-Datei zur Verfügung und können zu den PDOs (Process Data Objects) der rechten Seite hinzugefügt oder wieder entfernt werden.

Die [StandardDataTypes] können eingefügt werden, um im PDO leere Zwischenräume zu erzeugen.

PDO-Mapping: Einfügen

10046

Mit der Schaltfläche [Einfügen] können Sie weitere PDOs erzeugen und mit entsprechenden Objekten belegen. Über die eingefügten PDOs erfolgt die Zuordnung der Ein- und Ausgänge zu den IEC-Adressen.

In der Steuerungskonfiguration werden die vorgenommenen Einstellungen nach Verlassen des Dialoges sichtbar. Die einzelnen Objekte können dort mit symbolischen Namen belegt werden.

© ifm electronic gmbh

PDO-Mapping: Eigenschaften

10047

Über Eigenschaften lassen sich die in der Norm definierten Eigenschaften der PDOs in einem Dialog editieren:

COB-ID	Jede PDO-Nachricht benötigt einen eindeutigen COB-ID (Communication Object Identifier). Wird eine Option von dem Modul nicht unterstützt oder darf der Wert nicht verändert werden, so erscheint das Feld grau und kann nicht editiert werden.
Inhibit Time	Die Inhibit Time (100 µs) ist die minimale Zeit zwischen zwei Nachrichten dieses PDOs, damit die Nachrichten, die bei Änderung des Wertes übertragen werden, nicht zu häufig versendet werden. Die Einheit ist 100 µs.
Transmission Type	Bei Transmission Type erhalten Sie eine Auswahl von möglichen Übertragungmodi für dieses Modul: **acyclic – synchronous** Das PDO wird nach einer Änderung mit dem nächsten SYNC übertragen. **cyclic – synchronous** Das PDO wird synchron übertragen, wobei [Number of SYNCs] die Anzahl der Synchronisationsnachrichten angibt, die zwischen zwei Übertragungen dieses PDOs liegen. **asynchronous – device specific** Das PDO wird ereignisgesteuert, d.h. wenn sich der Wert ändert, übertragen. Welche Daten auf diese Weise übertragen werden können, ist im Geräteprofil festgelegt. **asynchronous – manufacturer specific** Das PDO wird ereignisgesteuert, d.h. wenn sich der Wert ändert, übertragen. Welche Daten auf diese Weise übertragen werden, wird vom Gerätehersteller festgelegt. **(a)synchronous – RTR only** Diese Dienste sind nicht implementiert. **Number of SYNCs** Abhängig vom Transmission Type ist dieses Feld editierbar zur Eingabe der Anzahl der Synchronisationsnachrichten (Definition in [CAN-Parameter-Dialog], [Com. Cycle Period], [Sync Window Length], [Sync. COB-Id]), nach denen das PDO wieder versendet werden soll. **Event-Time** Abhängig vom Transmission Type wird hier die Zeitspanne in Millisekunden [ms] angegeben, die zwischen zwei Übertragungen des PDOs liegen soll.

Register [Service Data Objects]

1970

Index, Name, Wert, Typ und Default

Hier werden alle Objekte der EDS- oder DCF-Datei aufgelistet, die im Bereich von Index 2000_{16} bis $9FFF_{16}$ liegen und als beschreibbar definiert sind. Zu jedem Objekt werden Index, Name, Wert, Typ und Default angegeben. Der Wert kann verändert werden. Markieren Sie den Wert und drücken Sie die [Leertaste]. Nach Änderung können Sie den Wert durch die Taste [Eingabe] bestätigen oder mit [ESC] verwerfen.

Bei der Initialisierung des CAN-Buses werden die eingestellten Werte in Form von SDOs (Service Data Object) an die CAN-Module übertragen und haben damit direkten Einfluss auf das Objektverzeichnis des CANopen-Slaves. Sie werden im Normalfall bei jedem Start des Applikations-Programms neu geschrieben – unabhängig davon, ob sie im CANopen-Slave dauerhaft gespeichert werden.

© ifm electronic gmbh

Der Master zur Laufzeit

Inhalt:

Reset aller konfigurierten Slaves am Bus beim Systemstart 115
Abfrage des Slave-Gerätetyps 115
Konfiguration aller fehlerfrei detektierten Geräte 115
Automatische Konfiguration von Slaves 116
Start aller fehlerfrei konfigurierten Slaves 116
Zyklisches Senden der SYNC-Message 116
Nodeguarding mit Lifetime-Überwachung 116
Heartbeat vom Master an die Slaves 116
Empfangen von Emergency-Messages 116

8569

Hier lesen Sie über Funktionalität der CANopen-Master-Bibliotheken zur Laufzeit.

Die CANopen-Master-Bibliothek stellt der CoDeSys-Applikation implizite Dienste zur Verfügung, die für die meisten Applikationen ausreichend sind. Diese Dienste werden für den Anwender transparent integriert und stehen in der Applikation ohne zusätzliche Aufrufe zur Verfügung. In der nachfolgenden Beschreibung wird davon ausgegangen, dass Sie zur Nutzung der Netzwerkdiagnose-, Status- und EMCY-Funktionen die Bibliothek `ifm_CRnnnn_CANopenMaster_Vxxyyzz.LIB` manuell im Bibliotheksverwalter eingefügt haben.

Zu den Diensten der CANopen-Master-Bibliothek zählen:

Reset aller konfigurierten Slaves am Bus beim Systemstart

8570

Um die Slaves zurückzusetzen, wird standardmäßig das NMT-Kommando "Reset Remote Node" benutzt, explizit für jeden Slave einzeln. (NMT steht nach CANopen für **N**etwork **M**anagment. Die einzelnen Kommandos sind im CAN-Dokument DSP301 beschrieben.) Um Slaves mit weniger leistungsstarken CAN-Controllern nicht zu überlasten, ist es sinnvoll, die Slaves mit einem Kommando "All Remote Nodes" zurückzusetzen.
Der Dienst wird für **alle** konfigurierten Slaves ausgeführt mit *CANx_MASTER_STATUS* (→ Seite 147) mit GLOBAL_START=TRUE. Sollen die Slaves **einzeln** zurückgesetzt werden, muss dieser Eingang auf FALSE gesetzt werden.

Abfrage des Slave-Gerätetyps

8021

Abfrage des Slave-Gerätetyps mittels SDO (Abfrage des Objekts 1000_{16}) und Vergleich mit dem konfigurierten Slave-ID:

Fehlerstatus-Ausgabe für die Slaves, von denen ein falscher Gerätetyp empfangen wurde. Die Anfrage wird nach 0,5 s wiederholt, wenn:
- kein Gerätetyp wurde empfangen
- UND Slave wurde in der Konfiguration **nicht** als optional markiert
- UND Timeout ist **nicht** abgelaufen.

Konfiguration aller fehlerfrei detektierten Geräte

8022

Jedes SDO wird auf Antwort überwacht und wiederholt, wenn sich innerhalb der Überwachungszeit der Slave nicht meldet.

© ifm electronic gmbh

Automatische Konfiguration von Slaves

8023

Automatische Konfiguration von Slaves mittels SDOs bei laufendem Busbetrieb:
Voraussetzung: Der Slave hat sich mittels Bootup-Message beim Master angemeldet.

Start aller fehlerfrei konfigurierten Slaves

8574

Start aller fehlerfrei konfigurierten Slaves nach dem Ende der Konfiguration des betreffenden Slaves:

Zum Starten der Slaves wird normalerweise das NMT-Kommando "Start remote node" benutzt. Wie beim "Reset" kann dieses Kommando durch "Start All Remote Nodes" ersetzt werden.

Der Dienst ist mittels CANx_Master_STATUS mit GLOBAL_START=TRUE aufrufbar.

Zyklisches Senden der SYNC-Message

8025

Dieser Wert ist nur bei der Konfiguration einstellbar.

Nodeguarding mit Lifetime-Überwachung

8576

Nodeguarding mit Lifetime-Überwachung für jeden Slave einstellbar:

Der Fehlerstatus kann für max. 8 Slaves mittels *CANx_MASTER_STATUS* (→ Seite 147) mit ERROR_CONTROL=TRUE überwacht werden.

Wir empfehlen: Für aktuelle Geräte besser mit Heartbeat arbeiten, weil dann die Buslast niedriger ist.

Heartbeat vom Master an die Slaves

8577

Der Fehlerstatus kann für max. 8 Slaves mittels CANx_MASTER_STATUS mit ERROR_CONTROL=TRUE überwacht werden.

Empfangen von Emergency-Messages

8578

Empfangen von Emergency-Messages für jeden Slave mit Speicherung der zuletzt empfangenen Emergency-Messages:

Die Fehlernachrichten können mittels CANx_MASTER_STATUS mit EMERGENCY_OBJECT_SLAVES=TRUE ausgelesen werden.

Zusätzlich liefert der FB die zuletzt erzeugte EMCY-Message am Ausgang GET_EMERGENCY.

© ifm electronic gmbh

Netzwerk starten

1863

Hier lesen Sie über das Starten des CANopen-Netzwerks.

Nach einem Download des Projekts auf die Steuerung oder einem Reset der Applikation wird das CAN-Netz vom Master neu hochgefahren. Das geschieht immer in der gleichen Reihenfolge von Aktionen:

- Alle Slaves werden zurückgesetzt, außer wenn sie als [nicht initialisieren] im Konfigurator markiert sind. Das Zurücksetzen geschieht einzeln mit dem NMT-Kommando "Reset Node" (81_{16}), jeweils mit dem Node-ID des Slaves. Wurde mit *CANx_MASTER_STATUS* (→ Seite 147) das Flag GLOBAL_START gesetzt, wird zum Hochfahren des Netzes das Kommando einmal mit Node-ID 0 benutzt.
- Alle Slaves werden konfiguriert. Dazu wird zunächst das Objekt 1000_{16} des Slaves abgefragt.
  - Wenn der Slave innerhalb der Überwachungszeit von 0,5 Sekunden antwortet, wird das jeweils nächste Konfigurations-SDO gesendet.
  - Ist ein Slave als [optional] markiert und antwortet nicht innerhalb der Überwachungszeit auf die Abfrage des Objekts 1000_{16}, wird er als nicht vorhanden markiert und keine weiteren SDOs werden an ihn geschickt.
  - Wenn ein Slave auf die Abfrage des Objekts 1000_{16} mit einem anderen Typ als dem konfigurierten (in den unteren 16 Bit) antwortet, wird er zwar konfiguriert, aber als falscher Typ markiert.
- Alle SDOs werden jeweils solange wiederholt, bis innerhalb einer Überwachungszeit eine Antwort des Slaves gesehen wurde. Hier kann die Applikation den Hochlauf der einzelnen Slaves überwachen und ggf. durch Setzen des Flags SET_TIMEOUT_STATE im NODE_STATE_SLAVE-Array des FB CANx_MASTER_STATUS reagieren.
- Wenn der Master eine Heartbeat-Zeit ungleich 0 konfiguriert hat, beginnt die Erzeugung des Hearbeats sofort nach dem Starten der Mastersteuerung.
- Nachdem alle Slaves ihre Konfigurations-SDOs erhalten haben, beginnt für Slaves mit konfiguriertem Nodeguarding das Guarding.
- Wenn der Master auf [automatisch starten] konfiguriert wurde, werden jetzt alle Slaves einzeln vom Master gestartet. Dazu wird das NMT-Kommando "Start Remote Node" (01_{16}) benutzt. Wurde mittels CANx_MASTER_STATUS das Flag GLOBAL_START gesetzt, dann wird das Kommando mit Node-ID 0 genutzt und somit alle Slaves mit einem "Start all Nodes" gestartet.
- Es werden mindestens einmal alle konfigurierten TX-PDOs gesendet (für die Slaves sind das RX-PDOs).
- Wenn [automatisch starten] deaktiviert wurde, müssen die Slaves einzeln über das Flag START_NODE im NODE_STATE_SLAVE-Array oder über den Eingang GLOBAL_START von CANx_MASTER_STATUS gestartet werden.

© ifm electronic gmbh

Netzwerkzustände

Inhalt:

Hochlauf des CANopen-Masters 118
Hochlauf der CANopen-Slaves 120
Hochlauf des Netzwerks ohne [Automatisch starten] 121
Das Objektverzeichnis des CANopen-Masters 124

1864

Hier lesen Sie, wie Sie die Zustände des CANopen-Netzwerks interpretieren und darauf reagieren können.

Beim *Netzwerk starten* (→ Seite 117) des CANopen Netzwerks und während des Betriebs durchlaufen die einzelnen Funktionsblöcke der Bibliothek verschiedene Zustände.

HINWEIS

Im Monitorbetrieb (Online-Modus) von CoDeSys können Sie die Zustände des CAN-Netzwerkes in der globalen Variablenliste "Can Open implicit variables" einsehen. Dazu sind genaue Kenntnisse von CANopen und der Struktur der CoDeSys-CANopen-Bibliotheken notwendig.

Um den Zugriff zu erleichtern, steht Ihnen *CANx_MASTER_STATUS* (→ Seite 147) aus der Bibliothek `ifm_CRnnnn_CANopenMaster_Vxxyyzz.LIB` zur Verfügung.

Hochlauf des CANopen-Masters

1971

Während des Hochlaufs des CAN-Netzwerks durchläuft der Master verschiedene Zustände, die Sie über den Ausgang NODE_STATE des FB *CANx_MASTER_STATUS* (→ Seite 147)

ablesen können.
(Netzwerk-Status des Masters → nächstes Kapitel)

Immer, wenn ein Slave auf eine SDO-Anfrage (Upload oder Download) nicht antwortet, dann wird die Anfrage wiederholt. Der Master verlässt den Status 3, wie oben beschrieben, aber erst, wenn alle SDOs erfolgreich übertragen wurden. So kann also erkannt werden, ob ein Slave fehlt oder ob der Master nicht alle SDOs richtig empfangen kann. Dabei ist es für den Master unerheblich, ob ein Slave mit einer Bestätigung oder einem Abort antwortet. Für den Master ist nur von Interesse, ob er überhaupt eine Antwort empfangen hat.

Eine Ausnahme stellt ein als [optional] markierter Slave dar. Optionale Slaves werden nur einmal nach ihrem Objekt 1000_{16} gefragt. Wenn sie nicht innerhalb von 0,5 s antworten, wird der Slave vom Master zunächst ignoriert und der Master geht auch ohne weitere Reaktion dieses Slaves in Status 5.

© ifm electronic gmbh

NMT-Status für CANopen-Master

9964

Status hex	dez	Beschreibung
00	0	nicht definiert
01	1	Master wartet auf die Bootup-Nachricht des Slaves. ODER: Master wartet auf Ablauf der GuardTime.
02	2	- Master wartet 300 ms. - Master fordert das Objekt 1000_{16} an. - Danach wechselt der Master auf Status 3.
03	3	Der Master konfiguriert seine Slaves. Dazu sendet der Master an die Slaves der Reihe nach alle vom Konfigurator erzeugten SDOs: - Der Master sendet an den Slave ein SDO-Read-Request (Index 1000_{16}). - Die generierten SDOs werden in ein SDO-Array gepackt. - Der Slave kennt seine erste SDO und die Anzahl seiner SDOs.
05	5	Nachdem an alle Slaves die SDOs übertragen wurden, geht der Master in den Status 5 und bleibt in diesem Status. Status 5 ist für den Master der normale Betriebszustand.

Knoten-Status aus FB lesen:

verwendeter Funktionsblock	hier steht dieser Knoten-Status
CANx_MASTER_STATUS CANx_SLAVE_STATUS	Ausgang NODE_STATE
CANOPEN_GETSTATE	Ausgang NODESTATE

© ifm electronic gmbh

Hochlauf der CANopen-Slaves

1972

Die Status eines Slaves können Sie über das Array NODE_STATE_SLAVE des FB *CANx_MASTER_STATUS* (→ Seite 147) auslesen.
(Netzwerk-Status der Slaves → nächstes Kapitel)

NMT-Status für CANopen-Slave

9965

Status hex	dez	Beschreibung
FF	-1	Der Slave wird durch die NMT-Nachricht [Reset Node] zurückgesetzt und wechselt selbständig in den Status 1.
00	0	nicht definiert
01	1	Status = Warten auf BOOTUP Der Slave wechselt nach einer maximalen Zeit von 2 s oder sofort nach Empfang seiner Bootup-Message in den Status 2.
02	2	Status = BOOTUP Der Slave wechselt nach einer Verzögerungszeit von 0,5 s automatisch in den Status 3.
03	3	Status = PREPARED Im Status 3 wird der Slave konfiguriert. Der Slave bleibt solange im Status 3, bis er alle vom Konfigurator erzeugten SDOs erhalten hat. Dabei spielt es keine Rolle, ob während der Konfiguration vom Slave SDO-Transfers mit Abort (Fehler) oder ob alle fehlerfrei beantwortet wurden. Nur die vom Slave erhaltene Antwort als solche ist wichtig – nicht ihr Inhalt. Wenn im Konfigurator die Option [Knoten zurücksetzen] aktiviert wurde, wird nach dem Senden des Objekts 1011_{16} Subindex 1, der dann den Wert "load" enthält, ein erneuter Reset des Slaves durchgeführt. Der Slave wird dann wieder mit dem Upload des Objekts 1000_{16} angefragt. Slaves, bei denen während der Konfigurationsphase ein Problem auftritt, bleiben im Status 3 oder wechseln nach der Konfigurationsphase direkt in einen Fehlerstatus (Status > 5).
04	4	Status = PRE-OPERATIONAL Ein Knoten wechselt immer in den Status 4, außer: • es handelt sich um einen "optionalen" Slave und er wurde als nicht am Bus verfügbar detektiert (Abfrage Objekt 1000_{16}) ODER: • der Slave ist zwar vorhanden, aber hat auf die Abfrage des Objekts 1000_{16} mit einem anderen Typ in den unteren 16 Bits reagiert, als der Konfigurator erwartet hat.
05	5	Status = OPERATIONAL Im Status 5 findet der normale Datenaustausch statt: "Normal Operation". Wenn der Master auf [Automatisch starten] konfiguriert wurde, wird der Slave im Status 4 gestartet (d.h. es wird eine "Start Node"-NMT-Nachricht erzeugt) und der Slave wechselt automatisch nach Status 5. Wurde GLOBAL_START gesetzt, dann wird gewartet, bis sich alle Slaves im Status 4 befinden. Anschließend werden alle Slaves mit dem NMT-Kommando [Start All Nodes] gestartet.
61	97	Ein Knoten wechselt in den Status 97, wenn er optional ist (optionales Gerät in der CAN-Konfiguration) und nicht auf die SDO-Anfrage nach dem Objekt 1000_{16} reagiert hat. Wird der Slave zu einem späteren Zeitpunkt an das Netzwerk angeschlossen und erkannt, wird er automatisch gestartet. Dazu müssen Sie aber die Option [Automatisch starten] in den CAN-Parametern des Masters angewählt haben.
62	98	Ein Knoten wechselt in den Status 98, wenn der Gerätetyp (Objekt 1000_{16}) nicht dem konfigurierten Typ entspricht.
63	99	Im Falle eines Nodeguarding-Timeouts wird der Slave auf Status 99 gesetzt. Sobald der Slave wieder auf NodeGuard-Anfragen reagiert und die Option [Automatisch starten] eingeschaltet ist, wird er automatisch vom Master gestartet. Dabei wird der Knoten abhängig von seinem Status, der in der Antwort auf die Nodeguard-Anfragen enthalten ist, neu konfiguriert oder nur gestartet. Um den Slave manuell zu starten, genügt es, die Methode [NodeStart] zu benutzen.

© ifm electronic gmbh

Der Master sendet Nodeguard-Nachrichten an den Slave, ...
- wenn sich der Slave im Status 4 oder höher befindet UND
- wenn Nodeguarding konfiguriert wurde.

Knoten-Status aus FB lesen:

verwendeter Funktionsblock	hier steht dieser Knoten-Status
CANx_MASTER_STATUS CANx_SLAVE_STATUS	Ausgang NODE_STATE
CANOPEN_GETSTATE	Ausgang NODESTATE

CANopen-Status des Knotens

1973

Knotenstatus nach CANopen (mit diesen Werten wird der Status auch in den entsprechenden Nachrichten vom Knoten her codiert).

Status hex	dez	CANopen-Status	Beschreibung
00	0	BOOTUP	Knoten hat die BOOTUP-Nachricht erhalten.
04	4	PREPARED	Knoten wird per SDOs konfiguriert.
05	5	OPERATIONAL	Knoten nimmt am normalen Datenaustausch teil.
7F	127	PRE-OPERATIONAL	Knoten sendet keine Daten, ist aber vom Master konfigurierbar.

Wenn Nodeguarding aktiv: das höchstwertige Status-Bit wechselt (toggelt) von Nachricht zu Nachricht.

Knoten-Status aus FB lesen:

verwendeter Funktionsblock	hier steht dieser Knoten-Status
CANx_MASTER_STATUS CANx_SLAVE_STATUS	Strukturelement LAST_STATE aus dem Array NODE_STATE_SLAVE
CANOPEN_GETSTATE	Ausgang LASTNODESTATE

Hochlauf des Netzwerks ohne [Automatisch starten]

8583

Manchmal ist es notwendig, dass die Applikation den Zeitpunkt bestimmt, wann die CANopen-Slaves gestartet werden. Dazu müssen Sie die Option [Automatisch starten] des CANopen-Masters in der Konfiguration deaktivieren. Dann ist die Applikation für das Starten der Slaves zuständig.

© ifm electronic gmbh

Starten des Netzwerks mit GLOBAL_START

1974

In einem CAN-Netz mit vielen Teilnehmern (meist mehr als 8) kommt es häufig dazu, dass schnell aufeinanderfolgende NMT-Nachrichten nicht von allen (meist langsamen) IO-Knoten (z.B. CompactModule CR2013) erkannt werden. Das liegt daran, dass diese Knoten alle Nachrichten mit dem ID 0 mithören müssen. In zu schneller Folge gesendete NMT-Nachrichten überlasten den Empfangspuffer solcher Knoten.

Eine Abhilfe können Sie schaffen, wenn die Anzahl schnell aufeinanderfolgender NMT-Nachrichten reduziert wird.

► Dazu von *CANx_MASTER_STATUS* (→ Seite 147) den Eingang GLOBAL_START auf TRUE setzen (mit [Automatisch starten]).

> Die CANopen-Master-Bibliothek benutzt den Befehl "Start All Nodes", anstatt alle Knoten einzeln mit dem Kommando "Start Node" zu starten.

> GLOBAL_START wird nur einmalig bei der Netzwerk-Initialisierung ausgeführt.

> Wenn dieser Eingang gesetzt wird, startet die Steuerung auch Knoten mit dem Status 98 (siehe oben). Die PDOs für diese Nodes bleiben jedoch weiterhin deaktiviert.

Starten des Netzwerks mit START_ALL_NODES

1975

Wird das Netzwerk nicht automatisch mit GLOBAL_START des FB *CANx_MASTER_STATUS* (→ Seite 147) gestartet, kann es jederzeit gestartet werden, d.h. jeder Knoten einzeln nacheinander. Ist das nicht gewünscht, besteht folgende Möglichkeit:

► Von CANx_MASTER_STATUS den Eingang START_ALL_NODES auf TRUE setzen. START_ALL_NODES wird typisch zur Laufzeit durch das Applikations-Programm gesetzt.

> Wenn dieser Eingang gesetzt wird, werden auch Knoten mit dem Status 98 (siehe oben) gestartet. Die PDOs für diese Nodes bleiben jedoch weiterhin deaktiviert.

Initialisieren des Netzwerks mit RESET_ALL_NODES

1976

Aus denselben Gründen, die für den Befehl START_ALL_NODES sprechen, gibt es Fälle, in denen Sie besser das NMT-Kommando RESET_ALL_NODES (anstelle RESET_NODES für jeden einzelnen Knoten) einsetzen.

► Dazu müssen Sie von *CANx_MASTER_STATUS* (→ Seite 147) den Eingang RESET_ALL_NODES auf TRUE setzen.

> Dadurch werden einmalig alle Knoten gleichzeitig zurückgesetzt.

© ifm electronic gmbh

Zugriff auf den Status des CANopen-Masters

1977

Damit der Applikations-Code erst abgearbeitet wird, wenn das IO-Netzwerk bereit ist, sollten Sie den Status des Masters abfragen. Das folgende Code-Fragment-Beispiel zeigt eine Möglichkeit:

Variablendeklaration

```
VAR
    FB_MasterStatus := CR0020_MASTER_STATUS;
    :
END_VAR
```

Programmcode

```
IF   FB_MasterStatus.NODE_STATE = 5 THEN
     <Applikationscode>
END_IF
```

Durch Setzen des Flags TIME_OUT_STATE im Array NODE_STATE_SLAVE des FB *CANx_MASTER_STATUS* (→ Seite 147) kann die Applikation reagieren und zum Beispiel den nicht konfigurierbaren Knoten überspringen.

© ifm electronic gmbh

Das Objektverzeichnis des CANopen-Masters

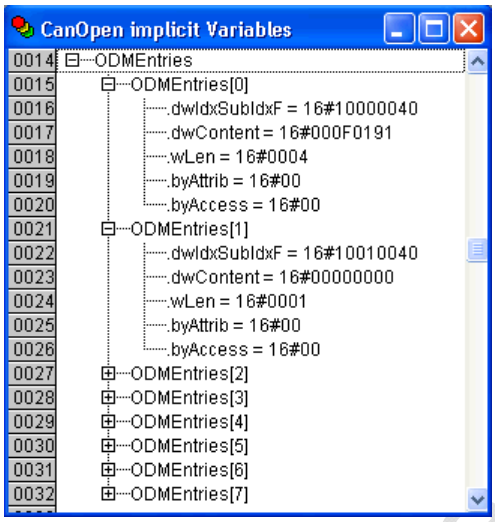
1978

In manchen Fällen ist es hilfreich, wenn der CANopen-Master über ein eigenes Objektverzeichnis verfügt. Das ermöglicht z.B. den Datenaustausch der Applikation mit anderen CAN-Knoten.

Das Objektverzeichnis des Masters wird über eine EDS-Datei mit dem Namen `CRnnnnMasterODEntry.EDS` während der Übersetzungszeit erstellt und mit Werten vorbelegt. Diese EDS-Datei ist im Verzeichnis `CoDeSys Vn\Library\PLCconf` abgelegt. Der Inhalt der EDS-Datei kann über die Schaltfläche [EDS...] im Konfigurations-Fenster [CAN-Parameter] angesehen werden.

Auch, wenn das Objektverzeichnis nicht vorhanden ist, kann der Master ohne Einschränkungen genutzt werden.

Der Zugriff auf das Objektverzeichnis durch die Applikation erfolgt über ein Array, das die folgende Struktur hat:



Strukturelement	Beschreibung
.dwIdxSubIdxF	Die Struktur der Komponente iiiissff$_{16}$ ist: iiii - Index (2 Byte, Bits 16...31), Idx ss - Subindex (1 Byte, Bits 8...15), SubIdx ff - Flags (1 Byte, Bits 0...7), F Die Flag-Bits haben folgende Bedeutung: Bit 0 = Schreiben (Write) Bit 1 = Inhalt ist ein Zeiger auf eine Adresse (Content is pointer) Bit 2 = mapbar (mappable) Bit 3 = swap Bit 4 = Vorzeichen behafteter Wert (signed) Bit 5 = Fließkomma (float) Bit 6 = Weitere Subindizes enthalten (has more elements)
.dwContent	Inhalt des Eintrags
.wLen	Länge der Daten
.byAttrib	Ursprünglich als Zugriffsberechtigung gedacht. Kann von der Applikation des Masters beliebig genutzt werden.
.byAccess	Früher Zugriffsberechtigung. Kann von der Applikation des Masters beliebig genutzt werden.

An der Oberfläche verfügt CoDeSys über keinen Editor für dieses Objektverzeichnis.

© ifm electronic gmbh

Die EDS-Datei gibt nur vor, mit welchen Objekten das Objektverzeichnis angelegt wird. Dabei werden die Einträge immer mit der Länge 4 erzeugt und die Flags (niederwertigstes Byte der Komponente eines Objektverzeichniseintrags `.dwIdxSubIdxF`) immer mit 1 belegt. D.h. beide Bytes werden mit 41_{16} belegt.

Wenn ein Objektverzeichnis im Master vorhanden ist, kann der Master als SDO-Server im Netz auftreten. Immer wenn ein Client auf einen Objektverzeichnis-Eintrag schreibend zugreift, wird das der Applikation über das Flag OD_CHANGED in *CANx_MASTER_STATUS* (→ Seite 147) angezeigt. Nach der Auswertung müssen Sie dieses Flag wieder zurücksetzen.

Die Applikation kann das Objektverzeichnis nutzen, indem die Einträge direkt beschrieben oder gelesen werden, oder indem die Einträge auf IEC-Variablen zeigen. D.h.: beim Lesen/Schreiben eines anderen Knotens wird direkt auf diese IEC-Variablen zugegriffen.

Wenn Index und Subindex des Objektverzeichnisses bekannt sind, kann ein Eintrag wie folgt angesprochen werden:

```
I := GetODMEntryValue(16#iiiiss00, pCanOpenMaster[0].wODMFirstIdx,
pCanOpenMaster[0].wODMFirstIdx + pCanOpenMaster[0].wODMCount;
```

Wobei für "iiii" der Index und für "ss" der Subindex (als Hex-Werte) eingesetzt werden müssen.

Damit steht die Nummer des Array-Eintrags in I zur Verfügung. Nun können Sie direkt auf die Komponenten des Eintrags zugreifen.
Damit Sie diesen Eintrag direkt auf einer IEC-Variable ausgeben können, genügt es, Adresse, Länge und Flags einzutragen:

```
ODMEntries[I].dwContent := ADR(<Variablenname>);
ODMEntries[I].wLen := sizeof(<Variablenname>);
ODMEntries[I].dwIdxSubIdxF := ODMEntries[I].dwIdxSubIdxF OR
OD_ENTRYFLG_WRITE OR OD_ENTRYFLG_ISPOINTER;
```

Um nur den Inhalt des Eintrags zu ändern, genügt es, den Inhalt von "`.dwContent`" zu ändern.

© ifm electronic gmbh

CANopen-Slave

Inhalt:

Funktionalität der CANopen-Slave-Bibliothek 126
CANopen-Slave konfigurieren 127
Zugriff auf den CANopen-Slave zur Laufzeit 134

1865

Eine CoDeSys-programmierbare Steuerung kann in einem CAN-Netzwerk auch als CANopen-Slave erscheinen.

Funktionalität der CANopen-Slave-Bibliothek

1979

Die CANopen-Slave-Bibliothek zusammen mit dem CANopen-Konfigurator stellt dem Anwender folgende Möglichkeiten zur Verfügung:

- In CoDeSys: Konfiguration der Eigenschaften NodeGuarding/Heartbeat, Emergency, Node-ID und Baudrate, auf der das Device arbeiten soll.
- Zusammen mit dem Parametermanager in CoDeSys kann ein Default-PDO-Mapping erstellt werden, das zur Laufzeit vom Master geändert werden kann. Die Änderung des PDO-Mappings erfolgt während der Konfigurationsphase durch den Master. Durch das Mapping können IEC-Variablen der Applikation in PDOs gemappt werden. D.h. den PDOs werden IEC-Variable zugeordnet, um sie im Applikations-Programm einfach auswerten zu können.
- Die CANopen-Slave-Bibliothek stellt ein Objektverzeichnis zur Verfügung. Die Größe dieses Objektverzeichnisses wird zur Übersetzungszeit von CoDeSys festgelegt. In diesem Verzeichnis befinden sich alle Objekte, die den CANopen-Slave beschreiben und zusätzlich die, die vom Parametermanager definiert sind. Im Parametermanager können zusammen mit dem CANopen-Slave nur die Listenarten Parameter und Variablen verwendet werden.
- Die Bibliothek verwaltet die Zugriffe auf das Objektverzeichnis, tritt also am Bus als SDO-Server auf.
- Die Bibliothek überwacht das Nodeguarding und die Heartbeat-Consumer-Zeit (immer nur von einem Producer) und setzt entsprechende Fehlerflags für die Applikation.
- Es kann eine EDS-Datei erzeugt werden, die die konfigurierten Eigenschaften des CANopen-Slaves so beschreibt, dass das Device als Slave unter einem CANopen-Master eingebunden und konfiguriert werden kann.

Die CANopen-Slave-Bibliothek stellt ausdrücklich folgende, in CANopen beschriebene, Funktionalitäten nicht zur Verfügung (alle hier und im obigen Abschnitt nicht genannten Möglichkeiten des CANopen-Protokolls sind ebenfalls nicht implementiert):

- Dynamische SDO- und PDO-Identifier
- SDO Block-Transfer
- Automatische Erzeugung von Emergency-Nachrichten. Emergency-Nachrichten müssen immer mittels *CANx_SLAVE_EMCY_HANDLER* (→ Seite 155) und *CANx_SLAVE_SEND_EMERGENCY* (→ Seite 157) von der Applikation erzeugt werden. Die Bibliothek `ifm_CRnnnn_CANopenSlave_Vxxyyzz.LIB` stellt Ihnen dazu diese FBs zur Verfügung.
- Dynamische Änderungen der PDO-Eigenschaften werden z.Z. immer nur beim Eintreffen einer StartNode NMT-Nachricht übernommen, nicht mit den in CANopen definierten Mechanismen.

© ifm electronic gmbh

CANopen-Slave konfigurieren

Inhalt:

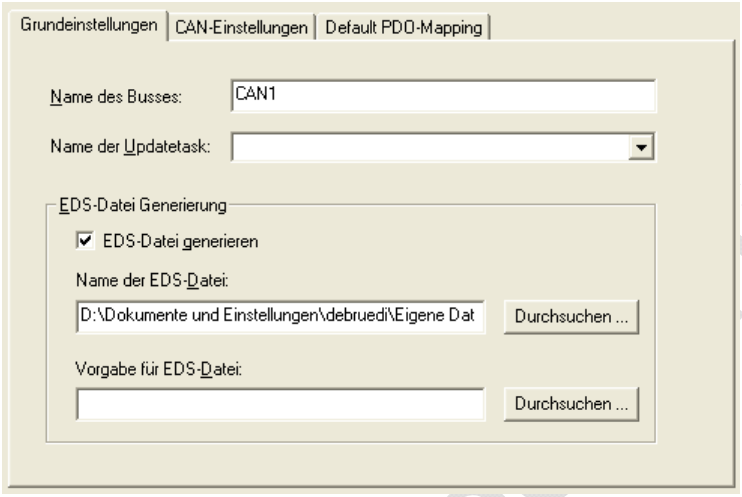
Register [Grundeinstellungen] 127
Register [CAN-Einstellungen] 130
Register [Default PDO-Mapping] 131
Verändern des Standard-Mappings durch Master-Konfiguration 133

1980

Um die Steuerung als CANopen-Slave zu nutzen, muss zunächst in der Steuerungskonfiguration über [Einfügen] > [Unterelement anhängen] der CANopen-Slave eingefügt werden. Bei Steuerungen mit 2 oder mehr CAN-Schnittstellen wird automatisch CAN-Schnittstelle 1 als Slave konfiguriert. Alle notwendigen Bibliotheken werden automatisch in den Bibliotheksverwalter eingefügt.

Register [Grundeinstellungen]

1981



Grundeinstellungen: Name des Busses

10049

Parameter wird im Moment nicht benutzt.

Grundeinstellungen: Name der Updatetask

10050

Name der Task, in der der Aufruf des CANopen-Slave erfolgt.

Grundeinstellungen: EDS-Datei generieren

10051

Soll aus den Einstellungen hier eine EDS-Datei erzeugt werden, um den CANopen-Slave in eine beliebigen Masterkonfiguration einfügen zu können, muss hier die Option [EDS-Datei generieren] aktiviert werden und der Name einer Datei angegeben werden. Optional kann auch noch eine Vorlagendatei angeben werden, deren Einträge zum EDS-File des CANopen-Slave hinzugefügt werden. Bei Überschneidungen werden Vorgaben der Vorlage nicht überschrieben.

© ifm electronic gmbh

Beispiel für ein Objektverzeichnis

1991

Folgende Einträge könnten zum Beispiel im Objektverzeichnis stehen:

```
[FileInfo]
FileName=D:\CoDeSys\lib2\plcconf\MyTest.eds
FileVersion=1
FileRevision=1
Description=EDS for CoDeSys-Project:
D:\CoDeSys\CANopenTestprojekte\TestHeartbeatODsettings_Device.pro
CreationTime=13:59
CreationDate=09-07-2005
CreatedBy=CoDeSys
ModificationTime=13:59
ModificationDate=09-07-2005
ModifiedBy=CoDeSys

[DeviceInfo]
VendorName=3S Smart Software Solutions GmbH
ProductName=TestHeartbeatODsettings_Device
ProductNumber=0x33535F44
ProductVersion=1
ProductRevision=1
OrderCode=xxxx.yyyy.zzzz
LMT_ManufacturerName=3S GmbH
LMT_ProductName=3S_Dev
BaudRate_10=1
BaudRate_20=1
BaudRate_50=1
BaudRate_100=1
BaudRate_125=1
BaudRate_250=1
BaudRate_500=1
BaudRate_800=1
BaudRate_1000=1
SimpleBootUpMaster=1
SimpleBootUpSlave=0
ExtendedBootUpMaster=1
ExtendedBootUpSlave=0

...

[1018sub0]
ParameterName=Number of entries
ObjectType=0x7
DataType=0x5
AccessType=ro
DefaultValue=2
PDOMapping=0

[1018sub1]
ParameterName=VendorID
ObjectType=0x7
DataType=0x7
AccessType=ro
DefaultValue=0x0
PDOMapping=0

[1018sub2]
ParameterName=Product Code
ObjectType=0x7
DataType=0x7
AccessType=ro
DefaultValue=0x0
PDOMapping=0
```

Bedeutung der einzelnen Objekte entnehmen Sie bitte der CANopen-Spezifikation DS301.

© ifm electronic gmbh

Die EDS-Datei enthält, neben den vorgeschriebenen Einträgen, die Definitionen für SYNC, Guarding, Emergency und Heartbeat. Wenn diese Objekte nicht benutzt werden, sind die Werte auf 0 gesetzt (voreingestellt). Da die Objekte aber im Objektverzeichnis des Slaves zur Laufzeit vorhanden sind, werden sie in der EDS-Datei auch beschrieben.

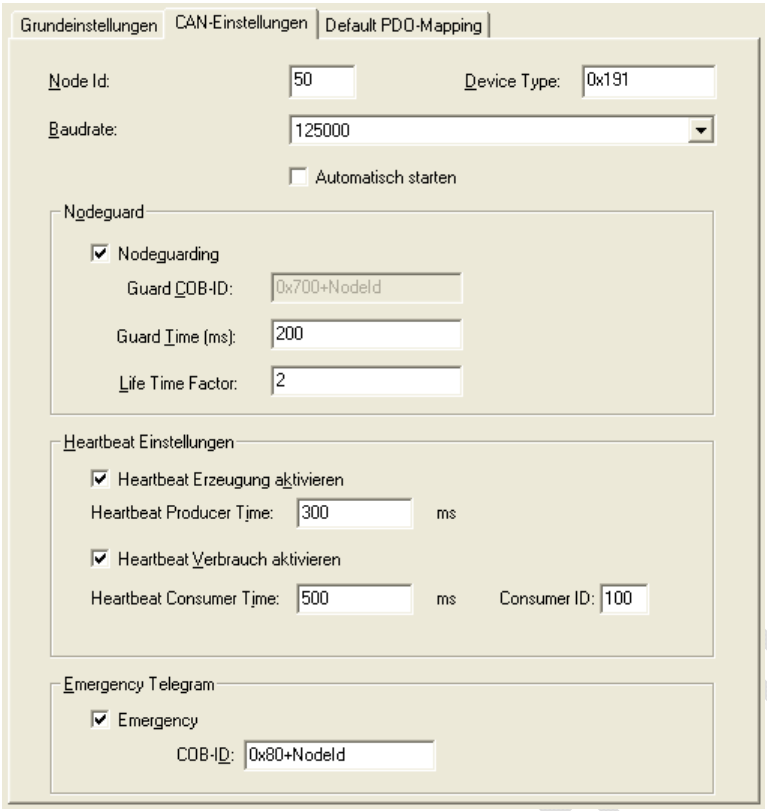
Das Gleiche gilt für die Einträge für die Kommunikations- und Mapping-Parameter. Es sind immer alle 8 möglichen Subindizes der Mapping-Objekte $16xx_{16}$ oder $1Axx_{16}$ vorhanden, aber u.U. im Subindex 0 nicht berücksichtigt.

Bit-Mapping wird von der Bibliothek nicht unterstützt!

© ifm electronic gmbh

Register [CAN-Einstellungen]

1982



Hier können Sie den **Node-ID** und die **Baudrate** einstellen.

Device Type
(das ist der Default-Wert des Objekts 1000_{16}, der im EDS eingetragen wird) wird mit 191_{16} (Standard-IO-Device) vorbelegt und kann von Ihnen beliebig geändert werden.
Der Index des CAN-Controllers ergibt sich aus der Position des CANopen-Slave in der Steuerungskonfiguration.

Die **Nodeguarding**-Parameter, die **Heartbeat**-Parameter und den Emergency-COB-ID können Sie ebenfalls auf diesem Register festlegen. Der CANopen-Slave kann nur für die Überwachung eines Heartbeats konfiguriert werden.
Wir empfehlen: Für aktuelle Geräte besser mit Heartbeat arbeiten, weil dann die Buslast niedriger ist.

HINWEIS

Beim Verwenden von Guarding oder Heartbeat UND
wenn Sie ein EDS-File erzeugen, das bei einem CANopen-Master eingebunden werden soll:

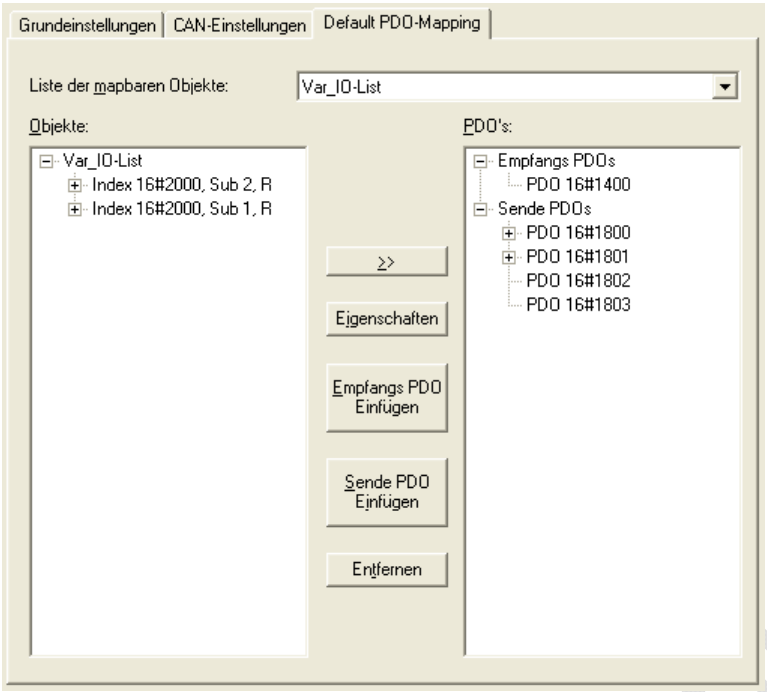
► Guard Time = 0 eintragen
Life Time Factor = 0 eintragen
Heartbeat Time = 0 eintragen

> Die beim CANopen-Master eingestellten Werte werden während der Konfiguration zum CANopen-Slave gesendet. Dadurch hat der CANopen-Master das Guarding oder den Heartbeat für diesen Knoten sicher aktiviert.

© ifm electronic gmbh

Register [Default PDO-Mapping]

1983



In diesem Register können Sie die Zuordnung zwischen lokalem Objektverzeichnis (OD-Editor) und den PDOs festlegen, die vom CANopen-Slave gesendet/empfangen werden. Eine solche Zuordnung wird als "Mapping" bezeichnet.

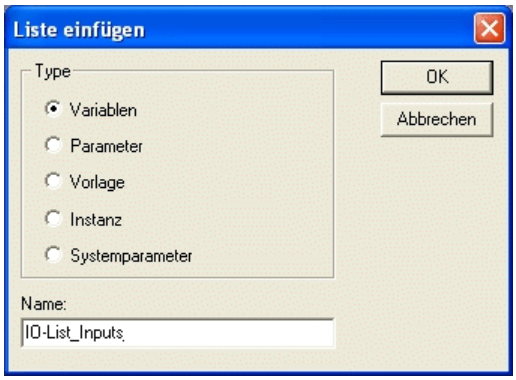
In den verwendeten Objektverzeichniseinträgen (Variablen OD) wird zwischen Objektindex/Subindex die Verbindung zu Variablen der Applikation hergestellt. Dabei müssen Sie nur darauf achten, dass der Subindex 0 eines Indexes, der mehr als einen Subindex enthält, die Information über die Anzahl der Subindizes enthält.

© ifm electronic gmbh

Beispiel: Variablenliste

10052

Auf dem ersten Empfangs-PDO (COB-ID = 512 + Node-ID) des CANopen-Slaves sollen die Daten für die Variable PLC_PRG.a empfangen werden.



Info

Als Listentyp kann [Variablen] oder [Parameter] gewählt werden.

Zum Datenaustausch (z.B. über PDOs oder sonstige Einträge im Objektverzeichnis) wird eine Variablenliste angelegt.

Die Parameterliste sollten Sie einsetzen, wenn Sie Objektverzeichniseinträge nicht mit Applikations-Variablen verknüpfen wollen. Für die Parameterliste ist zurzeit nur der Index 1006_{16} / SubIdx 0 vordefiniert. In diesen Eintrag kann vom Master der Wert für die [Com. Cycle Period] eingetragen werden. Damit wird das Ausbleiben der SYNC-Nachricht gemeldet.

Also müssen Sie im Objektverzeichnis (Parametermanager) eine Variablenliste anlegen und einen Index/SubIndex mit der Variablen PLC_PRG.a verknüpfen:

- ► Dazu fügen Sie in der Variablenliste eine Zeile hinzu (rechte Maustaste öffnet das Kontextmenü) und tragen einen Variablen-Namen (beliebig) sowie den Index und den Subindex ein.
- ► Als Zugriffsrichtung ist für ein Empfangs-PDO nur [write only] (schreiben) zugelassen.
- ► In die Spalte [Variable] tragen Sie dann "`PLC_PRG.a`" ein, oder drücken [F2] und wählen die Variable aus.

HINWEIS

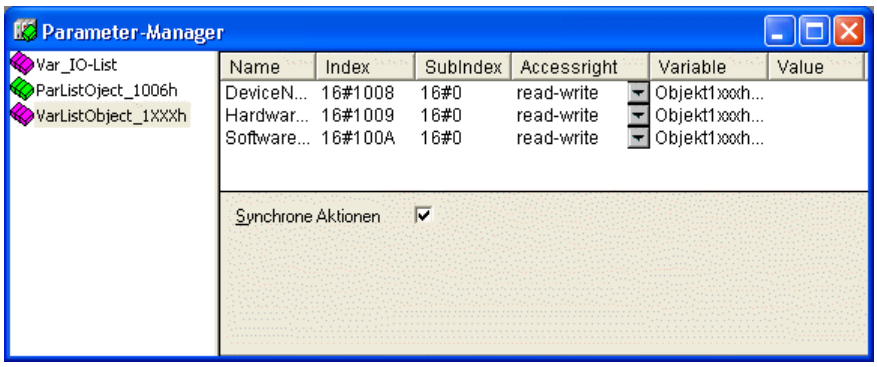
Daten, die vom CANopen-Master gelesen werden sollen (z.B. Eingänge, Systemvariablem), müssen die Zugriffsrichtung (Accessright) [read only] (lesen) haben.

Daten, die vom CANopen-Master geschrieben werden sollen (z.B. Ausgänge im Slave), müssen die Zugriffsrichtung (Accessright) [write only] (schreiben) haben.

SDO-Parameter, die vom CANopen-Master geschrieben und gleichzeitig aus der Slave-Applikation gelesen und geschrieben werden sollen, müssen die Zugriffsrichtung (Accessright) [read-write] (lesen+schreiben) haben.

© ifm electronic gmbh

Damit Sie den Parametermanager öffnen können, muss in den Zielsystemeinstellungen unter [Netzfunktionen] der Parametermanager aktiviert sein. Die Bereiche für Index/Subindex sind bereits mit sinnvollen Werten vorbelegt und sollten nicht geändert werden.



Im Default PDO-Mapping des CANopen-Slaves wird anschließend der Index-/Subindex-Eintrag als Mapping-Eintrag einem Empfangs-PDO zugewiesen. Die Eigenschaften des PDOs lassen sich über den Dialog festlegen, der aus Kapitel *CANopen-Slaves einfügen und konfigurieren* (→ Seite 111) bekannt ist.

Nur Objekte aus dem Parametermanager, die mit dem Attribut [read only] (lesen) oder [write only] (schreiben) versehen sind, werden in der evtl. erzeugten EDS-Datei als mapbar (= zuordnungsfähig) markiert und tauchen in der Liste der mapbaren Objekte auf. Alle anderen Objekte werden in der EDS-Datei als nicht mapbar markiert.

HINWEIS

Werden mehr als 8 Datenbytes in ein PDO gemappt, werden automatisch die nächsten freien Identifier dafür genutzt, bis alle Datenbytes übertragen werden können.

Um eine klare Struktur der verwendeten Identifier zu erhalten, sollten Sie die richtige Zahl der Empfangs- und Sende-PDOs einfügen und diesen die Variablen-Bytes aus der Liste zuordnen.

Verändern des Standard-Mappings durch Master-Konfiguration

1984

Sie können das vorgegebene PDO-Mapping (in der CANopen-Slave-Konfiguration) in bestimmten Grenzen durch den Master verändern.

Dabei gilt die Regel, dass der CANopen-Slave nicht in der Lage ist, Objektverzeichniseinträge neu anzulegen, die nicht bereits im Standard-Mapping (Default PDO-Mapping in der CANopen-Slave-Konfiguration) vorhanden sind. Also kann z.B. für ein PDO, das im Default PDO-Mapping ein gemapptes Objekt enthält, in der Masterkonfiguration kein zweites Objekt gemappt werden.

Das durch die Masterkonfiguration veränderte Mapping kann also höchstens die im Standard-Mapping vorhandenen PDOs enthalten. Innerhalb dieser PDOs sind 8 Mapping-Einträge (Subindizes) vorhanden.

Eventuelle Fehler, die hierbei auftreten können, werden Ihnen nicht angezeigt, d.h. die überzähligen PDO-Definitionen / die überzähligen Mapping-Einträge werden so behandelt, als seien sie nicht vorhanden.

Die PDOs müssen im Master immer von 1400_{16} (Empfangs-PDO-Kommunikationsparameter) oder 1800_{16} (Sende-PDO-Kommunikationsparameter) beginnend angelegt sein und lückenlos aufeinander folgen.

© ifm electronic gmbh

Zugriff auf den CANopen-Slave zur Laufzeit

1985

Einstellen der Knotennummer und der Baud-Rate eines CANopen-Slaves

1986

Beim CANopen-Slave kann zur Laufzeit des Applikations-Programms die Knotennummer und die Baudrate eingestellt werden.

- Zum Einstellen der **Knotennummer** wird *CANx_SLAVE_NODEID* (→ Seite 154) der Bibliothek `ifm_CRnnnn_CANopenSlave_Vxxyyzz.lib` genutzt.
- Zum Einstellen der **Baud-Rate** wird bei den Controllern und beim PDM360smart *CAN1_BAUDRATE* oder *CAN1_EXT* oder *CANx* der jeweiligen Gerätebibliothek benutzt. Beim PDM360 oder PDM360compact steht hierfür *CANx_SLAVE_BAUDRATE* (→ Seite 153) über die Bibliothek `ifm_CRnnnn_CANopenSlave_Vxxyyzz.lib` zur Verfügung.

Zugriff auf die OD-Einträge vom Applikations-Programm

1987

Standardmäßig gibt es Objektverzeichniseinträge, die auf Variablen gemappt sind (Parametermanager).

Es gibt jedoch auch die automatisch erzeugten Einträge des CANopen-Slave, auf die Sie nicht über den Parametermanager in einen Variableninhalt mappen können. Diese Einträge stehen mittels *CANx_SLAVE_STATUS* (→ Seite 160) in der Bibliothek `ifm_CRnnnn_CANopenSlave_Vxxyyzz.LIB` zur Verfügung.

Ändern der PDO-Eigenschaften zur Laufzeit

1988

Sollen die Eigenschaften eines PDOs zur Laufzeit verändert werden, so funktioniert das durch einen anderen Knoten über SDO-Schreibzugriffe, wie dies von CANopen beschrieben wird.

Alternativ kann man auch direkt eine neue Eigenschaft, wie z.B. die "Event time" eines Sende-PDOs schreiben und anschließend einen Befehl "StartNode-NMT" an den Knoten schicken, obwohl er bereits gestartet ist. Das führt dazu, dass das Device die Werte im Objektverzeichnis neu interpretiert.

Emergency-Messages durch das Applikations-Programm senden

1989

Um eine Emergency-Message durch das Applikations-Programm zu versenden, können Sie *CANx_SLAVE_EMCY_HANDLER* (→ Seite 155) und *CANx_SLAVE_SEND_EMERGENCY* (→ Seite 157) einsetzen. Die Bibliothek `ifm_CRnnnn_CANopenSlave_Vxxyyzz.LIB` stellt ihnen dazu diese FBs zur Verfügung.

© ifm electronic gmbh

CANopen-Netzwerkvariablen

Inhalt:

Allgemeine Informationen 135
CANopen-Netzwerkvariablen konfigurieren 136
Besonderheiten bei Netzwerkvariablen 140

1868

Allgemeine Informationen

2076

Netzwerkvariablen

CAN Netzwerkvariablen sind eine Möglichkeit, Daten zwischen zwei oder mehreren Steuerungen auszutauschen. Der Mechanismus sollte dabei für den Anwender möglichst einfach zu handhaben sein. Derzeit sind Netzwerkvariablen auf Basis von CAN und UDP implementiert. Die Variablenwerte werden dabei auf der Basis von Broadcast-Nachrichten automatisch ausgetauscht. In UDP sind diese als Broadcast-Telegramme realisiert, in CAN als PDOs. Diese Dienste sind vom Protokoll her nicht bestätigte Dienste, d.h. es gibt keine Kontrolle, ob die Nachricht auch beim Empfänger ankommt. Netzwerkvariablen-Austausch entspricht einer "1-zu-n-Verbindung" (1 Sender zu n Empfängern).

Objektverzeichnis

Das Objektverzeichnis ist eine weitere Möglichkeit, Variablen auszutauschen. Dabei handelt es sich um eine 1-zu-1-Verbindung, die ein bestätigtes Protokoll verwendet. Hier kann der Anwender also kontrollieren, ob die Nachricht den Empfänger erreichte. Der Austausch erfolgt nicht automatisch, sondern über den Aufruf von Funktionsblöcken aus dem Applikations-Programm.
→ Kapitel *Das Objektverzeichnis des CANopen-Masters*, → Seite 124

© ifm electronic gmbh

CANopen-Netzwerkvariablen konfigurieren

Inhalt:

Einstellungen in den Zielsystemeinstellungen 136
Einstellungen in den globalen Variablenlisten..... 137

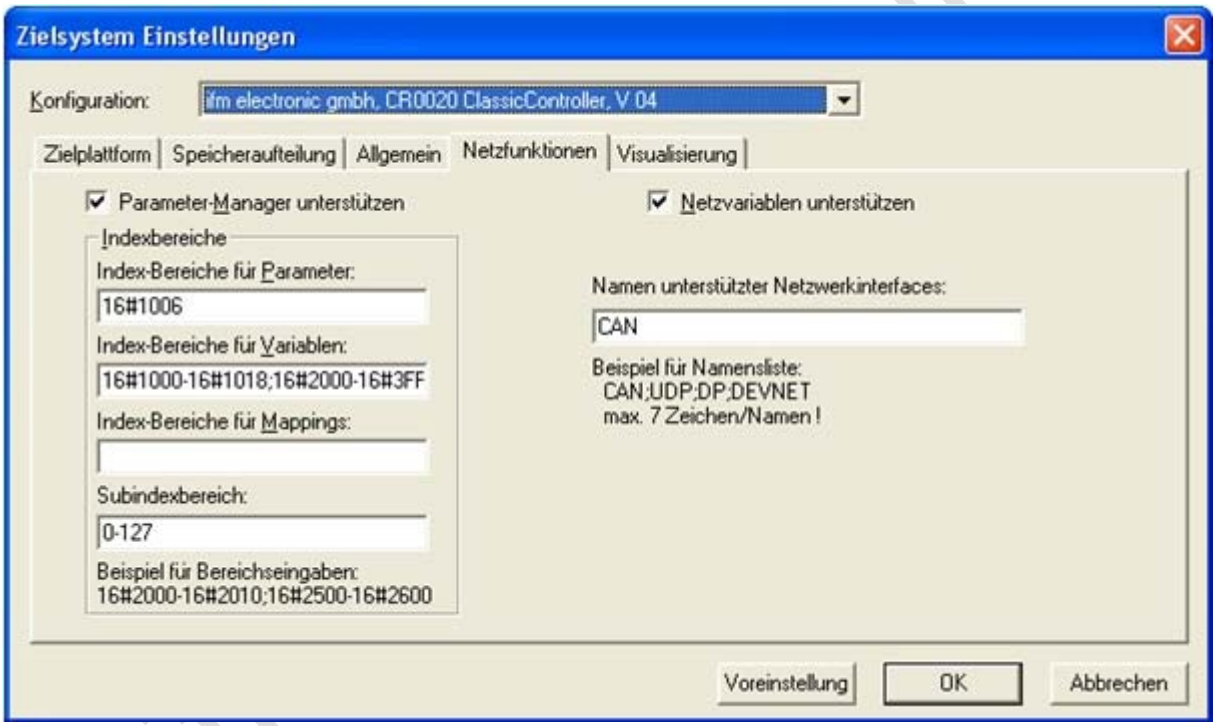
1869

Um die Netzwerkvariablen mit CoDeSys zu nutzen, benötigen Sie die folgenden Bibliotheken:
- `3s_CanDrv.lib`
- `3S_CANopenManager.lib`
- `3S_CANopenNetVar.lib`
- `SysLibCallback.lib`.

CoDeSys erzeugt automatisch den nötigen Initialisierungscode sowie den Aufruf der Netzwerk-Bausteine am Zyklusanfang und -ende.

Einstellungen in den Zielsystemeinstellungen

1994



Beispiel: Zielsystemeinstellungen für ClassicController CR0020

► Dialogbox [Zielsystemeinstellungen] wählen

► Register [Netzfunktionen] wählen

► Aktivieren Sie das Kontrollkästchen [Netzvariablen unterstützen].

► Bei [Namen unterstützter Netzwerkinterfaces] geben Sie den Namen des gewünschten Netzwerks an, hier CAN.

► Um Netzwerkvariablen zu nutzen, müssen Sie außerdem einen CANopen-Master oder CANopen-Slave in der Steuerungskonfiguration einfügen.

► Bitte beachten Sie die Besonderheiten bei der Anwendung von Netzwerkvariablen für die jeweiligen Gerätetypen
→ Kapitel *Besonderheiten bei Netzwerkvariablen*, → Seite 140

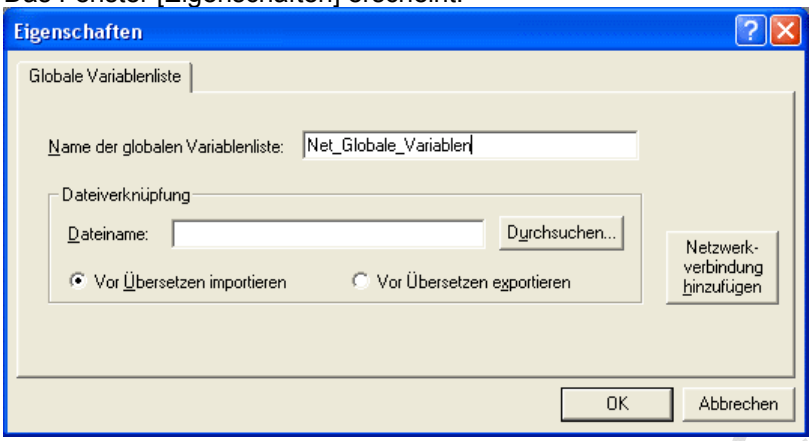
Einstellungen in den globalen Variablenlisten

1995

► Legen Sie eine neue globale Variablenliste an. Hier definieren Sie die Variablen, die sie mit anderen Steuerungen austauschen wollen.

► Öffnen Sie den Dialog mit dem Kontextmenü [Objekt Eigenschaften...].

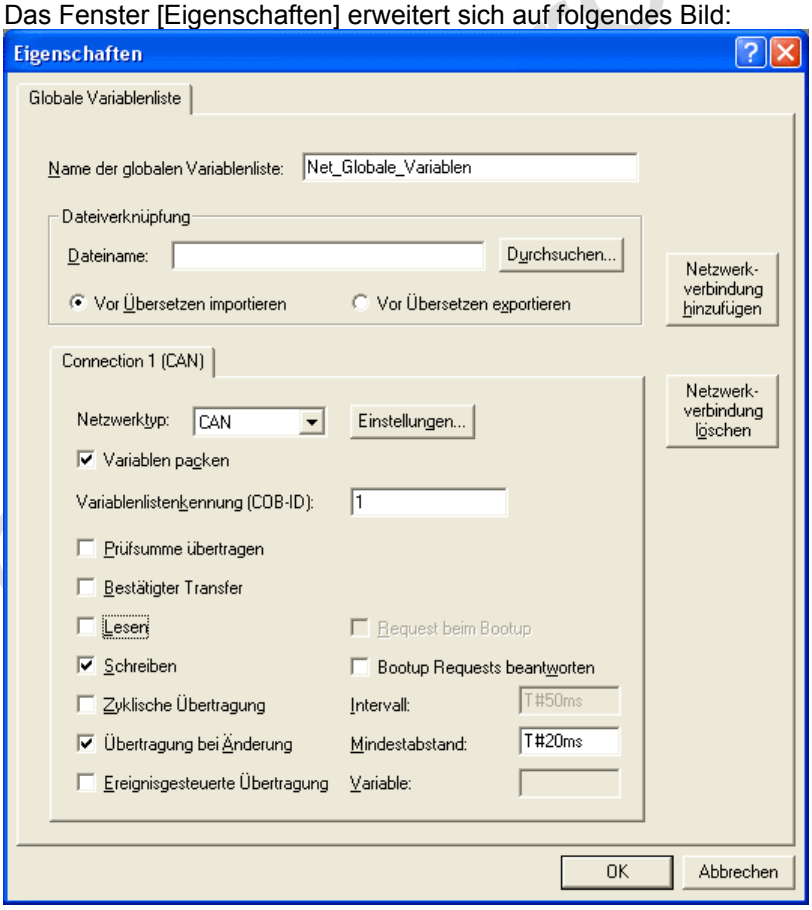
> Das Fenster [Eigenschaften] erscheint:



Wenn Sie die Netzwerkeigenschaften dieser Variablenliste definieren wollen:

► Schaltfläche [Netzwerkverbindung hinzufügen] klicken.
Wenn Sie mehrere Netzwerkverbindungen konfiguriert haben, können Sie hier auch pro Variablenliste mehrere Verbindungen konfigurieren.

> Das Fenster [Eigenschaften] erweitert sich auf folgendes Bild:



Die Optionen haben dabei folgende Bedeutungen:

Globale Variablenliste: Netzwerktyp

10055

Als Netzwerktyp können Sie einen der bei den Zielsystemeinstellungen angegebenen Netzwerknamen angeben.
Wenn Sie daneben auf die Schaltfläche [Einstellungen] klicken, können Sie die CAN-Schnittstelle wählen:

1. CAN-Schnittstelle: Wert = 0
2. CAN-Schnittstelle: Wert = 1

usw.

Globale Variablenliste: Variablen packen

10056

Wenn diese Option mit [**v**] aktiviert ist, werden die Variablen nach Möglichkeit in einer Übertragungseinheit zusammengefasst. Bei CAN ist eine Übertragungseinheit 8 Bytes groß.

Passen nicht alle Variablen der Liste in eine Übertragungseinheit, dann werden für diese Liste automatisch mehrere Übertragungseinheiten gebildet.

Ist die Option nicht aktiviert, kommt jede Variable in eine eigene Übertragungseinheit.

Wenn [Übertragung bei Änderung] konfiguriert ist, wird für jede Übertragungseinheit getrennt geprüft, ob sie geändert ist und gesendet werden muss.

Globale Variablenliste: Variablenlistenkennung (COB-ID)

10057

Der Basis-Identifier wird als eindeutige Kennung benutzt, um Variablenlisten verschiedener Projekte auszutauschen. Variablenlisten mit gleichem Basis-Identifier werden ausgetauscht. Es ist darauf zu achten, dass die Definitionen der Variablenlisten mit gleichem Basis-Identifier in den verschiedenen Projekten übereinstimmen.

HINWEIS

Der Basis-Identifier wird in CAN-Netzwerken direkt als COB-ID der CAN-Nachrichten benutzt. Es gibt keine Überprüfung, ob der Identifier auch in der übrigen CAN-Konfiguration benutzt wird.

Damit die Daten korrekt zwischen zwei Steuerungen ausgetauscht werden, müssen die globalen Variablenlisten in den beiden Projekten übereinstimmen. Sie können das Feature [Dateiverknüpfung] benutzen, um dies sicherzustellen. Ein Projekt kann die Variablenlisten-Datei vor dem Übersetzen exportieren. Die anderen Projekte sollten diese Datei vor dem Übersetzen importieren.

Neben einfachen Datentypen kann eine Variablenliste auch Strukturen und Arrays enthalten. Die Elemente dieser zusammengesetzten Datentypen werden einzeln versendet.

Es dürfen keine Strings über Netzwerkvariablen verschickt werden, da es sonst zu einem Laufzeitfehler kommt und der Watchdog aktiviert wird.

Wenn eine Variablenliste größer ist als ein PDO des entsprechenden Netzwerks, dann werden die Daten auf mehrere PDOs aufgeteilt. Es kann darum nicht zugesichert werden, dass alle Daten der Variablenliste in einem Zyklus empfangen werden. Teile der Variablenliste können in verschiedenen Zyklen empfangen werden. Dies ist auch für Variablen mit Struktur- und Array-Typen möglich.

© ifm electronic gmbh

Globale Variablenliste: Prüfsumme übertragen

10058

Diese Option wird nicht unterstützt.

Globale Variablenliste: Bestätigter Transfer

10059

Diese Option wird nicht unterstützt.

Globale Variablenliste: Lesen

10060

Es werden die Variablenwerte von einer (oder mehreren) Steuerungen gelesen.

Globale Variablenliste: Schreiben

10061

Die Variablen dieser Liste werden zu anderen Steuerungen gesendet.

HINWEIS

Sie sollten für jede Variablenliste nur eine dieser Möglichkeiten auswählen, also entweder nur lesen oder nur schreiben.

Wollen Sie verschiedene Variablen eines Projekts lesen und schreiben, so verwenden Sie bitte mehrere Variablenlisten (eine zum Lesen, eine zum Schreiben).

Für die Kommunikation zwischen 2 Teilnehmern sollten Sie die Variablenliste von einer Steuerung auf die andere kopieren, um die gleiche Datenstruktur zu erhalten.

Zwecks besserer Übersichtlichkeit sollten Ihre Variablenlisten jeweils nur für ein Teilnehmerpaar gelten. Es ist nicht sinnvoll, dieselbe Liste für alle Teilnehmer zu verwenden.

Globale Variablenliste: Zyklische Übertragung

10062

Nur gültig, wenn [Schreiben] aktiviert. Die Werte werden im angegebenen [Intervall] gesendet, unabhängig davon, ob sie sich geändert haben.

Globale Variablenliste: Übertragung bei Änderung

10063

Die Variablenwerte werden nur gesendet, wenn sich einer der Werte geändert hat. Mit [Mindestabstand] (Wert > 0) kann eine Mindestzeit zwischen den Nachrichtenpaketen festgelegt werden.

Globale Variablenliste: Ereignisgesteuerte Übertragung

10064

Wenn diese Option gewählt ist, wird die CAN-Nachricht nur dann übertragen, wenn die angegebene binäre [Variable] auf TRUE gesetzt wird. Diese Variable kann nicht über die Eingabehilfe aus der Liste der definierten Variablen gewählt werden.

© ifm electronic gmbh

Besonderheiten bei Netzwerkvariablen

1992

Gerät	Beschreibung
ClassicController: CR0020, CR0505 ExtendedController: CR0200 SafetyController: CR7020, CR7021, CR7200, CR7201, CR7505, CR7506	Netzwerkvariablen werden nur auf CAN-Schnittstelle 1 (Wert = 0 eintragen) unterstützt. **CANopen-Master** Sende- und Empfangslisten werden direkt verarbeitet. Sie brauchen nur die oben beschriebenen Einstellungen vornehmen. **CANopen-Slave** Sendelisten werden direkt verarbeitet. Für Empfangslisten müssen Sie zusätzlich noch den Bereich der Identifier im Objektverzeichnis auf Empfangs-PDOs mappen. Es ist ausreichend, wenn Sie nur zwei Empfangs-PDOs anlegen und dem ersten Objekt den ersten Identifier und dem zweiten Objekt den letzen Identifier zuweisen. Werden die Netzwerkvariablen nur auf einem Identifier übertragen, müssen Sie nur ein Empfangs-PDO mit diesem Identifier anlegen. Bitte beachten Sie, dass die Identifier der Netzwerkvariablen und der Empfangs-PDOs als **dezimale** Werte eingegeben werden müssen.
ClassicController: CR0032, CR0033 ExtendedController: CR0232, CR0233	Netzwerkvariablen werden auf allen CAN-Schnittstellen unterstützt. (Alle anderen Angaben wie oben)
BasicController: CR0403	Netzwerkvariablen werden auf allen CAN-Schnittstellen unterstützt. (Alle anderen Angaben wie oben)
BasicDisplay: CR0451	Es steht nur eine CAN-Schnittstelle zur Verfügung (Wert = 0 eintragen). (Alle anderen Angaben wie oben)
PDM360: CR1050, CR1051 PDM360compact: CR1052, CR1053, CR1055, CR1056 (nur CAN1 verfügbar)	Netzwerkvariablen werden auf den CAN-Schnittstellen 1 (Wert = 0) und 2 (Wert = 1) unterstützt. **CANopen-Master** Sende- und Empfangslisten werden direkt verarbeitet. Sie brauchen nur die oben beschriebenen Einstellungen vornehmen. **CANopen-Slave** Sende- und Empfangslisten werden direkt verarbeitet. Sie brauchen nur die oben beschriebenen Einstellungen vornehmen. Wird [Netzvariablen unterstützen] im PDM360 oder PDM360compact angewählt, müssen Sie mindestens eine Variable in der Globalen Varablenliste anlegen und diese einmalig im Applikations-Programm aufgerufen. Andernfalls wird die folgende Fehlermeldung bei der Programmübersetzung generiert: Fehler 4601: Netzwerkvariablen 'CAN' : Es ist keine zyklische oder freilaufende Task zum Netwerkvariablenaustausch vorhanden.
PDM360smart: CR1070, CR1071	Es steht nur eine CAN-Schnittstelle zur Verfügung (Wert = 0 eintragen). **CANopen-Master** Sende- und Empfangslisten werden direkt verarbeitet. Sie brauchen nur die oben beschriebenen Einstellungen vornehmen. **CANopen-Slave** Sendelisten werden direkt verarbeitet. Für Empfangslisten müssen Sie zusätzlich noch den Bereich der Identifier im Objektverzeichnis auf Empfangs-PDOs mappen. Es ist ausreichend, wenn Sie nur zwei Empfangs-PDOs anlegen und dem ersten Objekt den ersten Identifier und dem zweiten Objekt den letzen Identifier zuweisen. Werden die Netzwerkvariablen nur auf einem Identifier übertragen, müssen Sie nur ein Empfangs-PDO mit diesem Identifier anlegen. Bitte beachten Sie, dass die Identifier der Netzwerkvariablen und der Empfangs-PDOs als **dezimale** Werte eingegeben werden müssen.
PDM360NG: CR108n	Netzwerkvariablen werden auf allen CAN-Schnittstellen unterstützt. (Alle anderen Angaben wie oben)

© ifm electronic gmbh

7.5.2 Bibliotheken für CANopen

Inhalt:

ifm-Bibliothek für den CANopen-Master 141
ifm-Bibliothek für den CANopen-Slave 152
Weitere ifm-Bibliotheken zu CANopen 163

8587

ifm-Bibliothek für den CANopen-Master

Inhalt:

CANx_MASTER_EMCY_HANDLER 142
CANx_MASTER_SEND_EMERGENCY 144
CANx_MASTER_STATUS 147

1870

Für den CANopen-Master stellt die Bibliothek `ifm_CRnnnn_CANopenMaster_Vxxyyzz.LIB` eine Reihe von Bausteinen zur Verfügung, die im Folgenden erklärt werden.

© ifm electronic gmbh

CANx_MASTER_EMCY_HANDLER

2006

Baustein-Typ = Funktionsblock (FB)

x = Nr. 1...n der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Symbol in CoDeSys:

CAN1_MASTER_EMCY_HANDLER

9411

Enthalten in Bibliothek: `ifm_CRnnnn_CANopenMaster_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- CabinetController: CR030n
- ClassicController: CR0020, CR0032, CR0033, CR0505
- ExtendedController: CR0200, CR0232, CR0233
- Platinensteuerung: CS0015
- SafetyController: CR7nnn
- SmartController: CR25nn
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360smart: CR1070, CR1071

Beschreibung

2009

CANx_MASTER_EMCY_HANDLER überwacht den geräteeigenen Fehlerstatus des Masters. Der FB muss in folgenden Fällen aufgerufen werden:

- der Fehlerstatus soll ins Netzwerk übertragen werden und
- die Fehlermeldungen der Applikation sollen im Objektverzeichnis gespeichert werden.

Sollen applikations-spezifische Fehlernachrichten im Objektverzeichnis gespeichert werden, muss CANx_MASTER_EMCY_HANDLER **nach** dem (mehrfachen) Bearbeiten von *CANx_MASTER_SEND_EMERGENCY* (→ Seite 144) aufgerufen werden.

Parameter der Eingänge

2010

Parameter	Datentyp	Beschreibung
CLEAR_ERROR_FIELD	BOOL	TRUE: Löscht den Inhalt des Arrays ERROR_FIELD FALSE: diese Funktion wird nicht ausgeführt

© ifm electronic gmbh

Parameter der Ausgänge

2011

Parameter	Datentyp	Beschreibung
ERROR_REGISTER	BYTE	Zeigt den Inhalt des OBV Index 1001_{16} (Error Register)
ERROR_FIELD	ARRAY[0...5] OF WORD	Das Array[0...5] zeigt den Inhalt des OBV Index 1003_{16} (Error Field). ERROR_FIELD[0]: Anzahl der gespeicherten Fehler ERROR_FIELD[1...5]: gespeicherte Fehler, der jüngste Fehler steht im Index [1]

© ifm electronic gmbh

CANx_MASTER_SEND_EMERGENCY

2012

Baustein-Typ = Funktionsblock (FB)

x = Nr. 1...n der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Symbol in CoDeSys:

CAN1_MASTER_SEND_EMERGENCY

9430

Enthalten in Bibliothek: `ifm_CRnnnn_CANopenMaster_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- CabinetController: CR030n
- ClassicController: CR0020, CR0032, CR0033, CR0505
- ExtendedController: CR0200, CR0232, CR0233
- Platinensteuerung: CS0015
- SafetyController: CR7nnn
- SmartController: CR25nn
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360smart: CR1070, CR1071

Beschreibung

2015

CANx_MASTER_SEND_EMERGENCY versendet applikations-spezifische Fehlerstatus. Der FB wird aufgerufen, wenn der Fehlerstatus an andere Geräte im Netzwerkverbund übertragen werden soll.

Sollen applikations-spezifische Fehlernachrichten im Objektverzeichnis gespeichert werden, muss *CANx_MASTER_EMCY_HANDLER* (→ Seite 142) **nach** dem (mehrfachen) Bearbeiten von CANx_MASTER_SEND_EMERGENCY aufgerufen werden.

© ifm electronic gmbh

Parameter der Eingänge

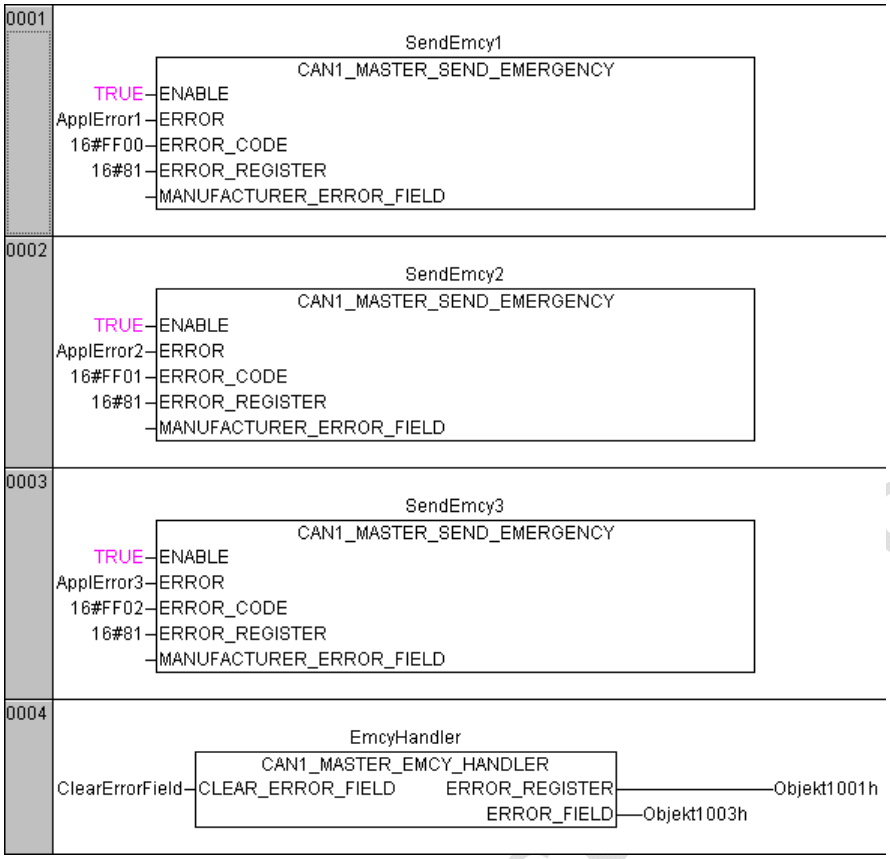
2016

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv
ERROR	BOOL	FALSE ⇨ TRUE (Flanke): sendet den anstehenden Fehlercode TRUE ⇨ FALSE (Flanke) UND Fehler steht nicht mehr an: Nach Verzögerung von ca. 1 s wird Null-Fehlermeldung gesendet sonst: diese Funktion wird nicht ausgeführt
ERROR_CODE	WORD	Der Error-Code gibt detailliert Auskunft über den erkannten Fehler. Die Werte sollten gemäß der CANopen-Spezifikation eingetragen werden. → Kapitel *Übersicht CANopen Error-Codes*, → Seite 175
ERROR_REGISTER	BYTE	Dieses Objekt spiegelt den allgemeinen Fehlerzustand des CANopen-Netzwerkteilnehmers wider. Die Werte sollten gemäß der CANopen-Spezifikation eingetragen werden.
MANUFACTURER_ERROR_FIELD	ARRAY[0...4] OF BYTE	Hier können bis zu 5 Bytes applikations-spezifische Fehlerinformationen eingetragen werden. Das Format ist dabei frei wählbar.

© ifm electronic gmbh

Beispiel: CANx_MASTER_SEND_EMERGENCY

2018



In diesem Beispiel werden nacheinander 3 Fehlermeldungen generiert:

1. ApplError1, Code = $FF00_{16}$ im Fehlerregister 81_{16}
2. ApplError2, Code = $FF01_{16}$ im Fehlerregister 81_{16}
3. ApplError3, Code = $FF02_{16}$ im Fehlerregister 81_{16}

Der FB CAN1_MASTER_EMCY_HANDLER sendet die Fehlermeldungen an das Fehler-Register "Objekt 1001_{16}" im Fehler-Array "Objekt 1003_{16}".

© ifm electronic gmbh

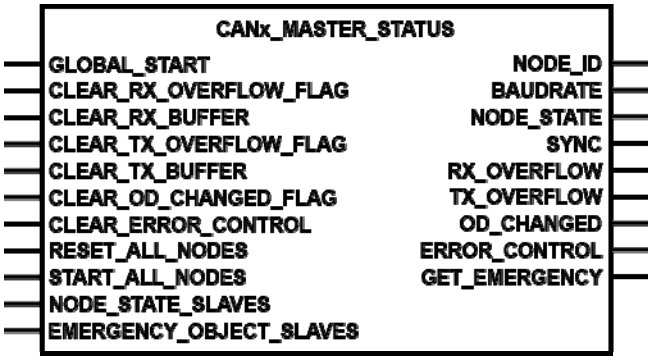
CANx_MASTER_STATUS

9933

Baustein-Typ = Funktionsblock (FB)

x = Nr. 1...n der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Symbol in CoDeSys:



CAN1_MASTER_STATUS

9435

Enthalten in Bibliothek:	Für folgende Geräte verfügbar:
`ifm_CRnnnn_CAN1openMaster_Vxxyyzz.LIB`	- PDM360: CR1050, CR1051
`ifm_CRnnnn_CANopenMaster_Vxxyyzz.LIB`	- PDM360compact: CR1052, CR1053, CR1055, CR1056 - PDM360smart: CR1070, CR1071

Beschreibung

2024

Status-Anzeige des als CANopen-Master eingesetzten Gerätes

Der FB zeigt den Status des als CANopen-Master eingesetzten Gerätes an. Außerdem kann der Status des Netzwerks und der angeschlossenen Slaves überwacht werden.

Der FB vereinfacht die Anwendung der CoDeSys-CANopen-Master-Bibliotheken. Wir empfehlen dringend, die Auswertung des Netzwerkstatus und der Fehlermeldungen über diesen FB vorzunehmen.

© ifm electronic gmbh

Parameter der Eingänge

2695

Parameter	Datentyp	Beschreibung
GLOBAL_START	BOOL	TRUE: Alle angeschlossenen Netzwerkteilnehmer (Slaves) werden gleichzeitig bei der Netzwerkinitialisierung gestartet. FALSE: Die angeschlossenen Netzwerkteilnehmer werden einzeln nacheinander gestartet. → Kapitel *Starten des Netzwerks mit GLOBAL_START*, → Seite 122
CLEAR_RX_OVERFLOW_FLAG	BOOL	FALSE ⇨ TRUE (Flanke): Fehlerflag "Empfangspuffer-Überlauf" löschen FALSE: diese Funktion wird nicht ausgeführt
CLEAR_RX_BUFFER	BOOL	FALSE ⇨ TRUE (Flanke): Daten im Empfangspuffer löschen FALSE: diese Funktion wird nicht ausgeführt
CLEAR_TX_OVERFLOW_FLAG	BOOL	FALSE ⇨ TRUE (Flanke): Fehlerflag "Sendepuffer-Überlauf" löschen FALSE: diese Funktion wird nicht ausgeführt
CLEAR_TX_BUFFER	BOOL	FALSE ⇨ TRUE (Flanke): Daten im Sendepuffer löschen FALSE: diese Funktion wird nicht ausgeführt
CLEAR_OD_CHANGED_FLAG	BOOL	FALSE ⇨ TRUE (Flanke): Flag "Daten im Objektverzeichnis geändert" löschen FALSE: diese Funktion wird nicht ausgeführt
CLEAR_ERROR_CONTROL	BOOL	FALSE ⇨ TRUE (Flanke): Die Guard-Fehlerliste (ERROR_CONTROL) löschen FALSE: diese Funktion wird nicht ausgeführt
RESET_ALL_NODES	BOOL	FALSE ⇨ TRUE (Flanke): Alle Knoten zurücksetzen FALSE: diese Funktion wird nicht ausgeführt
START_ALL_NODES	BOOL	TRUE: Alle angeschlossenen Netzwerkteilnehmer (Slaves) werden gleichzeitig zur Laufzeit des Applikations-Programms gestartet FALSE: Die angeschlossenen Netzwerkteilnehmer müssen einzeln nacheinander gestartet werden → Kapitel *Starten des Netzwerks mit START_ALL_NODES*, → Seite 122
NODE_STATE_SLAVES	DWORD	Zeigt den Status aller Netzwerkknoten. Beispiel-Code → Kapitel *Beispiel: CANx_MASTER_STATUS*, → Seite 150 → Kapitel *Der Master zur Laufzeit*, → Seite 115
EMERGENCY_OBJECT_SLAVES	DWORD	Zeigt die zuletzt aufgetretenen Fehlermeldungen aller Netzwerkknoten. → Kapitel *Zugriff auf die Strukturen zur Laufzeit der Applikation*, → Seite 151

© ifm electronic gmbh

Parameter der Ausgänge

9935

Parameter	Datentyp	Beschreibung
NODE_ID	BYTE	Node-ID des Masters
BAUDRATE	WORD	Baudrate des Masters
NODE_STATE	INT	aktueller Status des Masters.
SYNC	BOOL	SYNC-Signal des Masters. Dieses wird in Abhängigkeit der eingestellten Zeit Com. Cycle Period im *CANopen-Master: Register [CAN-Parameter]* (→ Seite 108) des Masters eingestellt.
RX_OVERFLOW	BOOL	Fehlerflag "Empfangspuffer-Überlauf"
TX_OVERFLOW	BOOL	Fehlerflag "Sendepuffer-Überlauf"
OD_CHANGED	BOOL	Flag "Objektverzeichnis Master wurde geändert"
ERROR_CONTROL	ARRAY [0..7] OF BYTE	Das Array enthält die Liste (max. 8) der fehlenden Netzwerkknoten (Guard- oder Heartbeat-Fehler) → Kapitel *Zugriff auf die Strukturen zur Laufzeit der Applikation*, → Seite 151
GET_EMERGENCY	STRUCT EMERGENY_MESSAGE	Am Ausgang stehen die Daten für die Struktur EMERGENY_MESSAGE zur Verfügung. Es wird immer die letzte Fehlermeldung eines Netzwerkknotens angezeigt. Um eine Liste aller aufgetretenen Fehler zu erhalten, muss das Array "EMERGENCY_OBJECT_SLAVES" ausgewertet werden.

Parameter der internen Strukturen

2698

Hier sehen Sie die Strukturen der in diesem Baustein genutzten Arrays.

Parameter	Datentyp	Beschreibung
CANx_EMERGENY_MESSAGE	STRUCT	NODE_ID: BYTE ERROR_CODE: WORD ERROR_REGISTER: BYTE MANUFACTURER_ERROR_FIELD: ARRAY[0..4] OF BYTE Die Struktur ist in den globalen Variablen der Bibliothek `ifm_CRnnnn_CANopenMaster_Vxxyyzz.LIB` angelegt.
CANx_NODE_STATE	STRUCT	NODE_ID: BYTE NODE_STATE: BYTE LAST_STATE: BYTE RESET_NODE: BOOL START_NODE: BOOL PREOP_NODE: BOOL SET_TIMEOUT_STATE: BOOL SET_NODE_STATE: BOOL Die Struktur ist in den globalen Variablen der Bibliothek `ifm_CRnnnn_CANopenMaster_Vxxyyzz.LIB` angelegt.

Ausführliche Beschreibung der Funktionalitäten des CANopen-Masters und der Mechanismen → Kapitel *CANopen-Master*, → Seite 105.

Die folgenden Code-Fragmente zeigen Ihnen am Beispiel des Controllers CR0032 die Anwendung des FB *CANx_MASTER_STATUS* (→ Seite 147).

© ifm electronic gmbh

Beispiel: CANx_MASTER_STATUS

2031

Slave-Informationen

2699

Damit Sie auf die Informationen der einzelnen CANopen-Knoten zugreifen können, müssen Sie ein Array über die jeweilige Struktur bilden. Die Strukturen sind in der Bibliothek enthalten. Sie können Sie im Bibliotheksverwalter unter [Datentypen] sehen.

Die Anzahl der Array-Elemente wird bestimmt durch die Globale Variable MAX_NODEINDEX, die automatisch vom CANopen-Stack angelegt wird. Sie enthält die Anzahl der im Netzwerkkonfigurator angegebenen Slaves minus 1.

Die Nummern der Array-Elemente entsprechen **nicht** dem Node-ID. Der Identifier kann aus der jeweiligen Struktur unter NODE_ID ausgelesen werden.

```
PROGRAM MasterStatus
VAR
    Status: CR0032_MASTER_STATUS;
    StartAllNodes: BOOL:= TRUE;
    ClearRxOverflowFlag: BOOL;
    ClearRxBuffer: BOOL;
    ClearTxOverflowFlag: BOOL;
    ClearTxBuffer: BOOL;
    ClearOdChanged: BOOL;
    ClearErrorControl: BOOL;
    ResetAllNodes: BOOL;
    ResetSingleNodeArray: ARRAY[0..MAX_NODEINDEX] OF RESET_NODE;
    NodeStateSlavesArray: ARRAY [0..MAX_NODEINDEX] OF NODE_STATE;
    EmergencyObjectSlavesArray: ARRAY[0..MAX_NODEINDEX] OF EMERGENCY_MESSAGE;
    node_id: BYTE;
    baudrate: WORD;
    node_state: INT;
    Sync: BOOL;
    RxOverflow: BOOL;
    TxOverflow: BOOL;
    OdChanged: BOOL;
    GuardHeartbeatErrorArray: ARRAY[0..7] OF BYTE;
    GetEmergency: EMERGENCY_MESSAGE;
END_VAR
```

Struktur Knoten-Status

2034

```
TYPE CAN1_NODE_STATE :
STRUCT
    NODE_ID: BYTE;
    NODE_STATE: BYTE;
    LAST_STATE: BYTE;
    RESET_NODE: BOOL;
    START_NODE: BOOL;
    PREOP_NODE: BOOL;
    SET_TIMEOUT_STATE: BOOL;
    SET_NODE_STATE: BOOL;
END_STRUCT
END_TYPE
```

Struktur Emergency_Message

2035

```
TYPE CAN1_EMERGENCY_MESSAGE :
STRUCT
    NODE_ID: BYTE;
    ERROR_CODE: WORD;
    ERROR_REGISTER: BYTE;
    MANUFACTURER_ERROR_FIELD: ARRAY[0..4] OF BYTE;
END_STRUCT
END_TYPE
```

Zugriff auf die Strukturen zur Laufzeit der Applikation

2036

Zur Laufzeit können Sie auf das jeweilige Array-Element über die globalen Variablen der Bibliothek zugreifen und so den Status oder die EMCY-Nachrichten auslesen oder den Knoten zurücksetzen.

```
0001 NodeStateList
0002     NodeStateList[0]
0003         .NODE_ID = 16#02
0004         .NODE_STATE = 16#04
0005         .LAST_STATE = 16#00
0006         .RESET_NODE = FALSE <:= TRUE>
0007         .START_NODE = FALSE
0008         .PREOP_NODE = FALSE
0009         .SET_TIMEOUT_STATE = FALSE
0010         .SET_NODE_STATE = FALSE
0011     NodeStateList[1]
0012         .NODE_ID = 16#03
0013         .NODE_STATE = 16#03
0014         .LAST_STATE = 16#00
0015         .RESET_NODE = FALSE
0016         .START_NODE = FALSE
0017         .PREOP_NODE = FALSE
0018         .SET_TIMEOUT_STATE = FALSE
0019         .SET_NODE_STATE = FALSE
0020 NodeEmergencyList
0021     NodeEmergencyList[0]
0022         .NODE_ID = 16#02
0023         .ERROR_CODE = 16#0000
0024         .ERROR_REGISTER = 16#00
0025         .MANUFACTURER_ERROR_FIELD
0026             .MANUFACTURER_ERROR_FIELD[0] = 16#00
0027             .MANUFACTURER_ERROR_FIELD[1] = 16#00
0028             .MANUFACTURER_ERROR_FIELD[2] = 16#00
0029             .MANUFACTURER_ERROR_FIELD[3] = 16#00
0030             .MANUFACTURER_ERROR_FIELD[4] = 16#00
0031     NodeEmergencyList[1]
0032         .NODE_ID = 16#03
0033         .ERROR_CODE = 16#0000
0034         .ERROR_REGISTER = 16#00
0035         .MANUFACTURER_ERROR_FIELD
0036             .MANUFACTURER_ERROR_FIELD[0] = 16#00
0037             .MANUFACTURER_ERROR_FIELD[1] = 16#00
0038             .MANUFACTURER_ERROR_FIELD[2] = 16#00
0039             .MANUFACTURER_ERROR_FIELD[3] = 16#00
0040             .MANUFACTURER_ERROR_FIELD[4] = 16#00
```

Setzen Sie im obigen Beispiel ResetSingleNodeArray[0].RESET_NODE kurzzeitig auf TRUE, wird der erste Knoten im Konfigurationsbaum zurückgesetzt.

ℹ zu den möglichen Fehler-Codes → Kapitel *CAN-Fehler und Fehlerbehandlung*, → Seite 170.

ifm-Bibliothek für den CANopen-Slave

Inhalt:

CANx_SLAVE_BAUDRATE 153
CANx_SLAVE_NODEID 154
CANx_SLAVE_EMCY_HANDLER 155
CANx_SLAVE_SEND_EMERGENCY 157
CANx_SLAVE_STATUS 160

1874

Für den CANopen-Slave stellt die Bibliothek `ifm_CRnnnn_CANopenSlave_Vxxyyzz.LIB` eine Reihe von Bausteinen zur Verfügung, die im Folgenden erklärt werden.

© ifm electronic gmbh

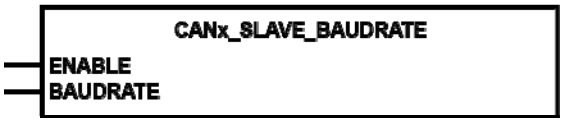
CANx_SLAVE_BAUDRATE

2038

Baustein-Typ = Funktionsblock (FB)

x = Nr. 1...n der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Symbol in CoDeSys:



CAN1_SLAVE_BAUDRATE

9489

Enthalten in Bibliothek: `ifm_CRnnnn_CANopenSlave_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Beschreibung

2041

CANx_SLAVE_BAUDRATE ermöglicht das Einstellen der Baudrate zur Laufzeit des Applikations-Programms.

Der FB wird im Normalfall bei der Initialisierung eines PDM360 Displays einmalig, im ersten Zyklus, aufgerufen. Anschließend wird der Eingang ENABLE wieder auf FALSE gesetzt.

HINWEIS

CANx_SLAVE_BAUDRATE wird nur in den Bibliotheken der Displays PDM360 und PDM360compact unterstützt.

Für die R360-Controller und das PDM360smart müssen Sie CAN1_BAUDRATE oder CAN1_EXT aus der jeweiligen Gerätebibliothek verwenden

Parameter der Eingänge

2042

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	FALSE ⇨ TRUE (Flanke): Baudrate setzen FALSE: Baustein wird nicht ausgeführt
BAUDRATE	BYTE	Wert der neuen Baudrate [kBit/s] Zulässige Werte: 10, 20, 50, 125, 250, 500, 800, 1000 Voreinstellwert → Datenblatt des Slaves

© ifm electronic gmbh

CANx_SLAVE_NODEID

2044

= CANx Slave Node-ID

Baustein-Typ = Funktionsblock (FB)

x = Nr. 1...n der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Symbol in CoDeSys:

CAN1_SLAVE_NODEID

9499

Enthalten in Bibliothek: `ifm_CRnnnn_CANopenSlave_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- CabinetController: CR030n
- ClassicController: CR0020, CR0032, CR0033, CR0505
- ExtendedController: CR0200, CR0232, CR0233
- Platinensteuerung: CS0015
- SafetyController: CR7nnn
- SmartController: CR25nn
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360smart: CR1070, CR1071

Beschreibung

2049

CANx_SLAVE_NODEID ermöglicht das Einstellen des Node-ID eines CANopen-Slaves zur Laufzeit des Applikations-Programms.

Der FB wird im Normalfall bei der Initialisierung der Steuerung einmalig, im ersten Zyklus, aufgerufen. Anschließend wird der Eingang ENABLE wieder auf FALSE gesetzt.

Parameter der Eingänge

2047

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	FALSE ⇨ TRUE (Flanke): NodeID setzen FALSE: Baustein wird nicht ausgeführt
NODEID	BYTE	Wert der neuen Knotennummer

© ifm electronic gmbh

CANx_SLAVE_EMCY_HANDLER

2050

Baustein-Typ = Funktionsblock (FB)

x = Nr. 1...n der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Symbol in CoDeSys:

CAN1_SLAVE_EMCY_HANDLER

9493

Enthalten in Bibliothek: `ifm_CRnnnn_CANopenSlave_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- CabinetController: CR030n
- ClassicController: CR0020, CR0032, CR0033, CR0505
- ExtendedController: CR0200, CR0232, CR0233
- Platinensteuerung: CS0015
- SafetyController: CR7nnn
- SmartController: CR25nn
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360smart: CR1070, CR1071

Beschreibung

2053

CANx_SLAVE_EMCY_HANDLER überwacht den geräteeigenen Fehlerstatus (Gerät wird als Slave betrieben).

Der FB muss in folgenden Fällen aufgerufen werden:

- der Fehlerstatus soll ins CAN-Netzwerk übertragen werden und
- die Fehlermeldungen der Applikation sollen im Objektverzeichnis gespeichert werden.

Sollen applikations-spezifische Fehlernachrichten im Objektverzeichnis gespeichert werden, muss CANx_SLAVE_EMCY_HANDLER **nach** dem (mehrfachen) Bearbeiten von *CANx_SLAVE_SEND_EMERGENCY* (→ Seite 157) aufgerufen werden.

Parameter der Eingänge

2054

Parameter	Datentyp	Beschreibung
CLEAR_ERROR_FIELD	BOOL	FALSE ⇨ TRUE (Flanke): ERROR-FIELD löschen FALSE: diese Funktion wird nicht ausgeführt

© ifm electronic gmbh

Parameter der Ausgänge

2055

Parameter	Datentyp	Beschreibung
ERROR_REGISTER	BYTE	Zeigt den Inhalt des OBV Index 1001_{16} (Error Register).
ERROR_FIELD	ARRAY[0...5] OF WORD	Das Array[0...5] zeigt den Inhalt des OBV Index 1003_{16} (Error Field): ERROR_FIELD[0]: Anzahl der gespeicherten Fehler ERROR_FIELD[1...5]: gespeicherte Fehler, der jüngste Fehler steht im Index [1].

© ifm electronic gmbh

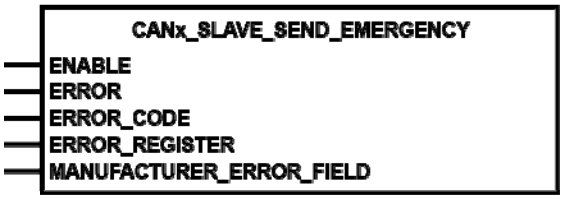
CANx_SLAVE_SEND_EMERGENCY

2056

Baustein-Typ = Funktionsblock (FB)

x = Nr. 1...n der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Symbol in CoDeSys:



CAN1_SLAVE_SEND_EMERGENCY

9505

Enthalten in Bibliothek: `ifm_CRnnnn_CANopenSlave_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- CabinetController: CR030n
- ClassicController: CR0020, CR0032, CR0033, CR0505
- ExtendedController: CR0200, CR0232, CR0233
- Platinensteuerung: CS0015
- SafetyController: CR7nnn
- SmartController: CR25nn
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360smart: CR1070, CR1071

Beschreibung

2059

Mit CANx_SLAVE_SEND_EMERGENCY werden applikations-spezifische Fehlerstatus versendet. Das sind Fehlernachrichten, die zusätzlich zu den geräteinternen Fehlernachrichten (z.B. Kurzschluss am Ausgang) gesendet werden sollen.

Der FB wird aufgerufen, wenn der Fehlerstatus an andere Geräte im Netzwerkverbund übertragen werden soll.

Sollen applikations-spezifische Fehlermachrichten im Objektverzeichnis gespeichert werden, muss *CANx_SLAVE_EMCY_HANDLER* (→ Seite 155) **nach** dem (mehrfachen) Bearbeiten von CANx_SLAVE_SEND_EMERGENCY aufgerufen werden.

© ifm electronic gmbh

Parameter der Eingänge

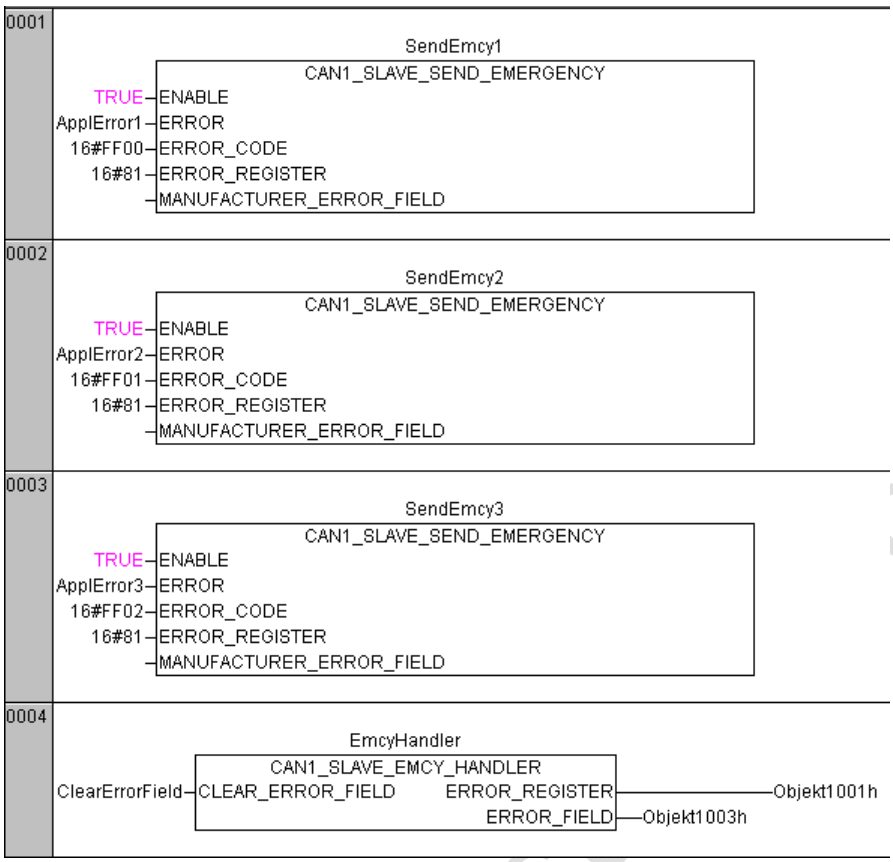
2060

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv
ERROR	BOOL	FALSE ⇨ TRUE (Flanke): sendet den anstehenden Fehlercode TRUE ⇨ FALSE (Flanke) UND Fehler steht nicht mehr an: Nach Verzögerung von ca. 1 s wird Null-Fehlermeldung gesendet sonst: diese Funktion wird nicht ausgeführt
ERROR_CODE	WORD	Der Error-Code gibt detailliert Auskunft über den erkannten Fehler. Die Werte sollten gemäß der CANopen-Spezifikation eingetragen werden. → Kapitel *Übersicht CANopen Error-Codes*, → Seite 175
ERROR_REGISTER	BYTE	Dieses Objekt spiegelt den allgemeinen Fehlerzustand des CANopen-Netzwerkteilnehmers wider. Die Werte sollten gemäß der CANopen-Spezifikation eingetragen werden.
MANUFACTURER_ERROR_FIELD	ARRAY[0...4] OF BYTE	Hier können bis zu 5 Bytes applikations-spezifische Fehlerinformationen eingetragen werden. Das Format ist dabei frei wählbar.

© ifm electronic gmbh

Beispiel: CANx_SLAVE_SEND_EMERGENCY

2062



In diesem Beispiel werden nacheinander 3 Fehlermeldungen generiert:

1. ApplError1, Code = $FF00_{16}$ im Fehlerregister 81_{16}
2. ApplError2, Code = $FF01_{16}$ im Fehlerregister 81_{16}
3. ApplError3, Code = $FF02_{16}$ im Fehlerregister 81_{16}

Der FB CAN1_SLAVE_EMCY_HANDLER sendet die Fehlermeldungen an das Fehler-Register "Objekt 1001_{16}" im Fehler-Array "Objekt 1003_{16}".

© ifm electronic gmbh

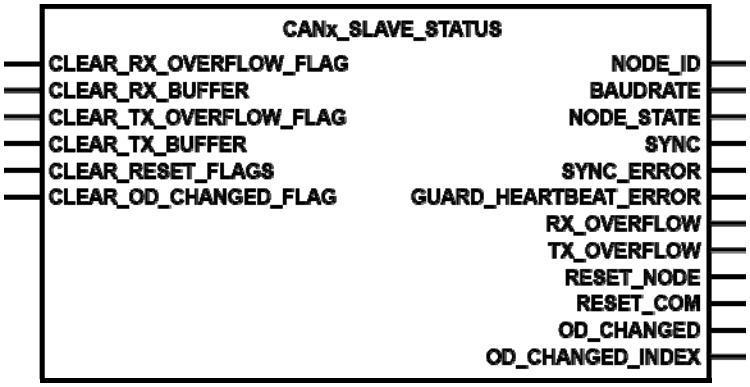
CANx_SLAVE_STATUS

2706

Baustein-Typ = Funktionsblock (FB)

x = Nr. 1...n der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Symbol in CoDeSys:



CAN1_SLAVE_STATUS

9510

Enthalten in Bibliothek:	Für folgende Geräte verfügbar:
`ifm_CRnnnn_CANopen1Slave_Vxxyyzz.LIB`	- ClassicController: CR0032, CR0033 - ExtendedController: CR0232, CR0233
`ifm_CRnnnn_CAN1openSlave_Vxxyyzz.LIB`	- PDM360: CR1050, CR1051
`ifm_CRnnnn_CANopenSlave_Vxxyyzz.LIB`	- PDM360compact: CR1052, CR1053, CR1055, CR1056 - PDM360smart: CR1070, CR1071

Beschreibung

2707

CANx_SLAVE_STATUS zeigt den Status des als CANopen-Slave eingesetzten Gerätes an. Der FB vereinfacht die Anwendung der CoDeSys-CANopen-Slave-Bibliotheken. Wir empfehlen dringend, die Auswertung des Netzwerkstatus über diesen FB vorzunehmen.

Eine ausführliche Beschreibung der Funktionalitäten des CANopen-Slaves und der Mechanismen → Kapitel *CANopen-Slave*, → Seite 126.

Zur Laufzeit können Sie dann auf die einzelnen Ausgänge des Bausteins zugreifen, um eine Statusübersicht zu erhalten.

© ifm electronic gmbh

Beispiel:

```
PROGRAM SlaveStatus
VAR
    Status: CR0032_SLAVE_STATUS;
    ClearRxOverflowFlag: BOOL;
    ClearRxBuffer: BOOL;
    ClearTxOverflowFlag: BOOL;
    ClearTxBuffer: BOOL;
    ClearResetFlag: BOOL;
    ClearOdChangedFlag: BOOL;
    node_id: BYTE;
    baudrate: WORD;
    node_state: BYTE;
    Sync: BOOL;
    SyncError: BOOL;
    GuardHeartbeatError: BOOL;
    RxOverflow: BOOL;
    TxOverflow: BOOL;
    ResetNode: BOOL;
    ResetCom: BOOL;
    OdChanged: BOOL;
    OdChangedIndex: INT;
END_VAR
```

Parameter der Eingänge

2708

Parameter	Datentyp	Beschreibung
CLEAR_RX_OVERFLOW_FLAG	BOOL	FALSE ⇨ TRUE (Flanke): Fehlerflag "Empfangspuffer-Überlauf" löschen FALSE: diese Funktion wird nicht ausgeführt
CLEAR_RX_BUFFER	BOOL	FALSE ⇨ TRUE (Flanke): Daten im Empfangspuffer löschen FALSE: diese Funktion wird nicht ausgeführt
CLEAR_TX_OVERFLOW_FLAG	BOOL	FALSE ⇨ TRUE (Flanke): Fehlerflag "Sendepuffer-Überlauf" löschen FALSE: diese Funktion wird nicht ausgeführt
CLEAR_TX_BUFFER	BOOL	FALSE ⇨ TRUE (Flanke): Daten im Sendepuffer löschen FALSE: diese Funktion wird nicht ausgeführt
CLEAR_RESET_FLAG	BOOL	FALSE ⇨ TRUE (Flanke): Flag "Knoten zurückgesetzt" löschen Flag "Kommunikationsschnittstelle zurückgesetzt" löschen FALSE: diese Funktion wird nicht ausgeführt
CLEAR_OD_CHANGED_FLAG	BOOL	FALSE ⇨ TRUE (Flanke): Flag "Daten im Objektverzeichnis geändert" löschen Flag "Index-Position" löschen FALSE: diese Funktion wird nicht ausgeführt

© ifm electronic gmbh

Parameter der Ausgänge

2068

Parameter	Datentyp	Beschreibung
NODE_ID	BYTE	Node-ID des Slaves
BAUDRATE	WORD	Baudrate des Slaves
NODE_STATE	BYTE	aktueller Status des Slaves
SYNC	BOOL	Empfangenes SYNC-Signal des Masters
SYNC_ERROR	BOOL	Es wurde kein SYNC-Signal des Masters empfangen ODER: die eingestellte SYNC-Zeit (ComCyclePeriod im Master) wurde überschritten.
GUARD_HEARTBEAT_ERROR	BOOL	Es wurde kein Guard- oder Heartbeat-Signal des Masters empfangen ODER: die eingestellten Zeiten wurden überschritten
RX_OVERFLOW	BOOL	Fehlerflag "Empfangspuffer-Überlauf"
TX_OVERFLOW	BOOL	Fehlerflag "Sendepuffer-Überlauf"
RESET_NODE	BOOL	Der CAN-Stack des Slaves wurde vom Master zurückgesetzt. Dieses Flag kann von der Applikation ausgewertet und ggf. für weitere Reaktionen genutzt werden.
RESET_COM	BOOL	Das Kommunikationsinterface des CAN-Stack wurde vom Master zuückgesetzt. Dieses Flag kann von der Applikation ausgewertet und ggf. für weitere Reaktionen genutzt werden.
OD_CHANGED	BOOL	Flag "Objektverzeichnis Master wurde geändert"
OD_CHANGED_INDEX	INT	Ausgang zeigt den geänderten Index des Objektverzeichnisses

© ifm electronic gmbh

Weitere ifm-Bibliotheken zu CANopen

Inhalt:

CANx_SDO_READ 164
CANx_SDO_WRITE 166
READ_COM_NODEID 168
SET_COM_NODED 169

2071

Hier stellen wir Ihnen weitere **ifm**-Bausteine vor, die für CANopen sinnvolle Ergänzungen darstellen.

© ifm electronic gmbh

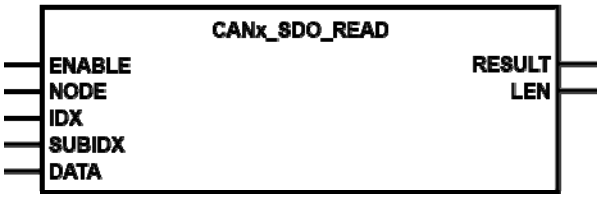
CANx_SDO_READ

621

Baustein-Typ = Funktionsblock (FB)

x = Nr. 1...n der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Symbol in CoDeSys:



CAN1_SDO_READ

9442

Enthalten in Bibliothek: `ifm_CANx_SDO_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Beschreibung

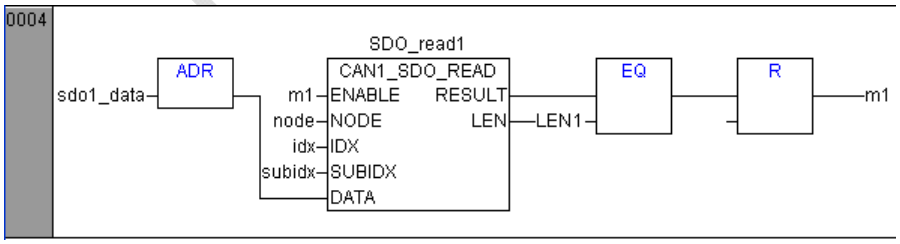
624

CANx_SDO_READ liest das SDO (→ Kapitel *Register [Service Data Objects]*, → Seite 114) mit den angegebenen Indizes aus dem Knoten aus.

Über diese können die Einträge im Objektverzeichnis gelesen werden. Dadurch ist es möglich, die Knotenparameter gezielt zu lesen.

- alle ecomatmobile-Controller - Platinensteuerung: CS0015 - PDM360smart: CR1070, CR1071	- PDM360: CR1050, CR1051 - PDM360compact: CR1052, CR1053, CR1055, CR1056
aus Gerätebibliothek `ifm_CRnnnn_Vxxyyzz.LIB`	aus Gerätebibliothek `ifm_CANx_SDO_Vxxyyzz.LIB`
Voraussetzung: Knoten muss sich im Zustand PRE-OPERATIONAL oder OPERATIONAL befinden.	Voraussetzung: Knoten muss sich im Modus "CANopen-Master" oder "CANopen-Slave" befinden.

Beispiel:



ifm electronic gmbh

Parameter der Eingänge

625

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv
NODE	BYTE	Nummer des Knotens
IDX	WORD	Index im Objektverzeichnis
SUBIDX	BYTE	Subindex bezogen auf den Index im Objektverzeichnis
DATA	DWORD	Adresse des Empfangsdaten-Arrays zulässige Länge = 0...255 Übergabe mit ADR-Operator

Parameter der Ausgänge

626

Parameter	Datentyp	Beschreibung
RESULT	BYTE	0 = Baustein inaktiv 1 = Baustein-Ausführung beendet 2 = Baustein ist aktiv 3 = Fehler: Baustein wurde nicht ausgeführt
LEN	WORD	Länge des Eintrags in "Anzahl der Bytes" Der Wert für LEN muss mit der Länge des Empfangs-Arrays übereinstimmen. Andernfalls treten Störungen bei der SDO-Kommunikation auf.

© ifm electronic gmbh

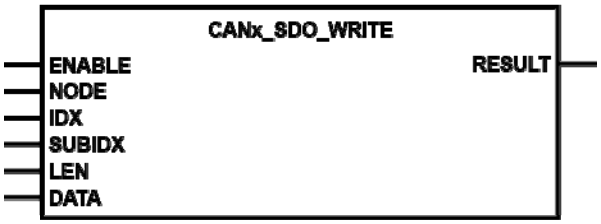
CANx_SDO_WRITE

615

Baustein-Typ = Funktionsblock (FB)

x = Nr. 1...n der CAN-Schnittstelle (je nach Gerät, → Datenblatt)

Symbol in CoDeSys:



CAN1_SDO_WRITE

9457

Enthalten in Bibliothek: `ifm_CANx_SDO_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Beschreibung

618

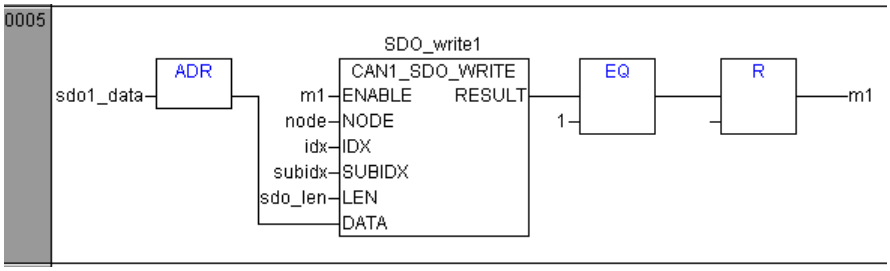
CANx_SDO_WRITE schreibt das SDO (→ Kapitel *Register [Service Data Objects]*, → Seite 114) mit den angegebenen Indizes in den Knoten.

Über diesen FB können die Einträge im Objektverzeichnis geschrieben werden. Dadurch ist es möglich, die Knotenparameter gezielt zu setzen.

- alle ecomatmobile-Controller - Platinensteuerung: CS0015 - PDM360smart: CR1070, CR1071	- PDM360: CR1050, CR1051 - PDM360compact: CR1052, CR1053, CR1055, CR1056
aus Gerätebibliothek `ifm_CRnnnn_Vxxyyzz.LIB`	aus Gerätebibliothek `ifm_CANx_SDO_Vxxyyzz.LIB`
Voraussetzung: Knoten muss sich im Zustand PRE-OPERATIONAL oder OPERATIONAL befinden und im Modus "CANopen-Master".	Voraussetzung: Knoten muss sich im Modus "CANopen-Master" oder "CANopen-Slave" befinden.

Der Wert für LEN muss mit der Länge des Sendearrays übereinstimmen. Andernfalls treten Störungen bei der SDO-Kommunikation auf.

Beispiel:



© ifm electronic gmbh

Parameter der Eingänge

619

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv
NODE	BYTE	Nummer des Knotens
IDX	WORD	Index im Objektverzeichnis
SUBIDX	BYTE	Subindex bezogen auf den Index im Objektverzeichnis.
LEN	WORD	Länge des Eintrags in "Anzahl der Bytes" Der Wert für LEN muss mit der Länge des Sende-Arrays übereinstimmen. Andernfalls treten Störungen bei der SDO-Kommunikation auf.
DATA	DWORD	Adresse des Sendedaten-Arrays zulässige Länge = 0...255 Übergabe mit ADR-Operator

Parameter der Ausgänge

620

Parameter	Datentyp	Beschreibung
RESULT	BYTE	0 = Baustein inaktiv 1 = Bausteinausführung beendet 2 = Baustein ist aktiv 3 = Fehler: Baustein wurde nicht ausgeführt

© ifm electronic gmbh

READ_COM_NODEID

3214

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_UTIL_Vxxyyzz.Lib`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Symbol in CoDeSys:

Beschreibung

3322

READ_COM_NODEID liest den aktuellen CAN-Download-ID.

Parameter der Eingänge

3323

Parameter	Datentyp	Beschreibung
READ	BOOL	Flanke FALSE ⇨ TRUE: liest den CAN-Download-ID FALSE: Baustein wird nicht ausgeführt

Parameter der Ausgänge

3324

Parameter	Datentyp	Beschreibung
READ_ID	STRING(3)	aktueller CAN-Download-ID

© ifm electronic gmbh

SET_COM_NODED

3217

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_UTIL_Vxxyyzz.Lib`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Symbol in CoDeSys:

Beschreibung

3325

SET_COM_NODED setzt den CAN-Download-ID auf einen neuen Wert (1...127).

HINWEIS

Der CAN-Download-ID des Geräts muss mit dem in CoDeSys eingestellten CAN-Download-ID übereinstimmen!

Im CAN-Netzwerk müssen die CAN-Download-IDs einmalig sein!

Parameter der Eingänge

3326

Parameter	Datentyp	Beschreibung
NODEID	BYTE	neuer CAN-Download-ID (1...127)
SET	BOOL	Flanke FALSE ⇨ TRUE: setzt den CAN-Download-ID FALSE: Baustein wird nicht ausgeführt

Parameter der Ausgänge

3327

Parameter	Datentyp	Beschreibung
NEW_NODEID	STRING(3)	neuer CAN-Download-ID

© ifm electronic gmbh

7.6 CAN-Fehler und Fehlerbehandlung

Inhalt:

CAN-Fehler 171
Aufbau einer EMCY-Nachricht 173
Übersicht CANopen Error-Codes 175

1171

Die hier beschriebenen Fehlermechanismen werden von dem im Controller integrierten CAN-Controller automatisch abgearbeitet. Der Anwender hat darauf keinen Einfluss. Der Anwender sollte (je nach Applikation) auf gemeldete Fehler in der Anwendersoftware reagieren.

Ziel der CAN-Fehler-Mechanismen ist es:

- Sicherstellung einheitlicher Datenobjekte im gesamten CAN-Netz
- Dauerhafte Funktionsfähigkeit des Netzes auch im Falle eines defekten CAN-Teilnehmers
- Unterscheidung zwischen zeitweiliger und dauerhafter Störung eines CAN-Teilnehmers
- Lokalisierung und Selbstabschaltung eines defekten Teilnehmers in 2 Stufen:
  - Fehlerpassiv (Error-passiv)
  - Trennen vom Bus (Bus-off)

  Dies ermöglicht einem zeitweilig gestörten Teilnehmer eine "Erholungspause".

Um dem interessierten Anwender einen Überblick über das Verhalten des CAN-Controllers im Fehlerfall zu geben, soll an dieser Stelle vereinfacht die Fehlerbehandlung beschrieben werden. Nach der Fehlererkennung werden die Informationen automatisch aufbereitet und stehen in der Anwendersoftware dem Programmierer als CAN-Fehler-Bits zur Verfügung.

© ifm electronic gmbh

7.6.1 CAN-Fehler

8589

Fehlertelegramm

1172

Erkennt ein Busteilnehmer eine Fehlerbedingung, so sendet er sofort ein Fehlerflag und veranlasst damit den Abbruch der Übertragung bzw. das Verwerfen der von anderen Teilnehmern schon empfangenen fehlerfreien Nachrichten. Dadurch wird sichergestellt, dass allen Teilnehmern fehlerfreie und einheitliche Daten zur Verfügung stehen. Da das Fehlerflag unmittelbar übertragen wird, kann im Gegensatz zu anderen Feldbussystemen (diese warten eine festgelegte Quittierungszeit ab) sofort mit der Wiederholung der gestörten Nachricht durch den Absender begonnen werden. Dies ist eines der wichtigsten Merkmale von CAN.

Eine der grundsätzlichen Problematiken der seriellen Datenübertragung ist, dass ein dauerhaft gestörter oder defekter Busteilnehmer das gesamte System blockieren kann. Gerade die Fehlerbehandlung bei CAN würde solche Gefahr fördern. Um diesen Fall auszuschließen, ist ein Mechanismus erforderlich, welcher den Defekt eines Teilnehmers erkennt und diesen Teilnehmer gegebenenfalls vom Bus abschaltet.

Fehlerzähler

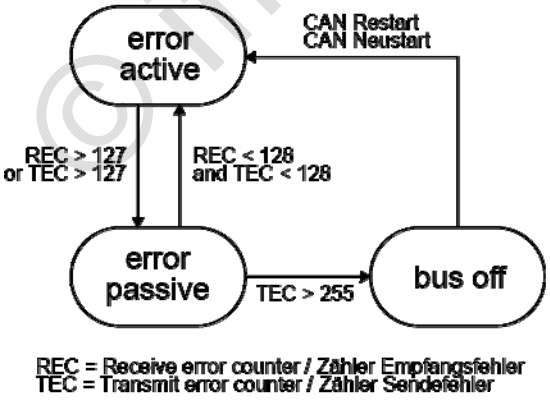
1173

Dazu sind im CAN-Controller ein Sende- und ein Empfangsfehlerzähler enthalten. Diese werden bei jedem fehlerhaften Sende- oder Empfangsvorgang heraufgezählt (inkrementiert). War eine Übertragung fehlerfrei, werden diese Zähler wieder heruntergezählt (dekrementiert).

Die Fehlerzähler werden jedoch im Fehlerfall stärker inkrementiert, als sie im Erfolgsfalle dekrementiert werden. Über eine bestimmte Zeitspanne kann dies zu einem merklichen Anstieg der Zählerstände führen, selbst wenn die Anzahl der ungestörten Nachrichten größer ist, als die Anzahl der gestörten Nachrichten. Längere fehlerfreie Zeitspannen bauen die Zählerstände langsam wieder ab. Die Zählerstände sind somit ein Maß für die relative Häufigkeit von gestörten Nachrichten.

Werden Fehler von einem Teilnehmer selbst als erster erkannt (= selbstverschuldete Fehler), wird bei diesem Teilnehmer der Fehler stärker "bestraft" als bei den anderen Busteilnehmern. Dazu wird der Zähler um einen höheren Betrag inkrementiert.

Übersteigt nun der Zählerstand eines Teilnehmers einen bestimmten Wert, kann davon ausgegangen werden, dass dieser Teilnehmer defekt ist. Damit dieser Teilnehmer den folgenden Busverkehr nicht weiter durch aktive Fehlermeldungen (error active) stört, wird er "fehlerpassiv" geschaltet (error passiv).



Grafik: Mechanismus des Fehlerzählers

error active
→ *Teilnehmer fehleraktiv*, → Seite 172

error passive
→ *Teilnehmer fehlerpassiv*, → Seite 172

bus off
→ *Teilnehmer bus-off*, → Seite 172

CAN Restart
→ Teilnehmer bus-off

©ifm electronic gmbh

Teilnehmer fehleraktiv

1174

Ein fehleraktiver Teilnehmer nimmt voll am Busverkehr teil und darf erkannte Fehler durch Senden des aktiven Fehlerflags signalisieren. Wie bereits beschrieben, wird dadurch die übertragene Nachricht zerstört.

Teilnehmer fehlerpassiv

1175

Ein fehlerpassiver Teilnehmer ist ebenfalls noch voll kommunikationsfähig. Er darf allerdings einen von ihm erkannten Fehler nur durch ein – den Busverkehr nicht störendes – passives Fehlerflag kenntlich machen. Ein fehlerpassiver Teilnehmer wird beim Unterschreiten eines festgelegten Zählerwertes wieder fehleraktiv.

Um den Anwender über das Ansteigen des Fehlerzählers zu informieren, wird bei einem Wert des Fehlerzählers $\geq$ 96 die Systemvariable CANx_WARNING gesetzt. Der Teilnehmer ist in diesem Zustand noch fehleraktiv.

Teilnehmer bus-off

1176

Wird der Fehlerzählerwert weiter inkrementiert, wird nach Überschreiten eines Maximalzählerwertes der Teilnehmer vom Bus abgeschaltet (bus-off).

Um diesen Zustand anzuzeigen, wird im Applikations-Programm der Merker CANx_BUSOFF gesetzt.

Der Fehler CANx_BUSOFF wird vom Betriebssystem automatisch behandelt und zurückgesetzt. Soll eine genauere Fehlerbehandlung und Auswertung über das Applikations-Programm erfolgen, muss *CANx_ERRORHANDLER* eingesetzt werden. Der Fehler CANx_BUSOFF muss dann explizit durch das Applikations-Programm zurückgesetzt werden.

© ifm electronic gmbh

7.6.2 Aufbau einer EMCY-Nachricht

Inhalt:

Man unterscheidet folgende Fehler: ... 173
Aufbau einer Fehlernachricht ... 173
Identifier ... 173
EMCY-Fehlercode ... 174
Objekt 0x1003 (Error Field) ... 174
Gerätefehler signalisieren ... 174

8591

Die Signalisierung von Fehlerzuständen erfolgt unter CANopen über einen sehr einfachen, standardisierten Mechanismus. Jedes Auftreten eines Fehlers bei einem CANopen-Gerät wird über eine spezielle Nachricht signalisiert, die den Fehler genauer beschreibt.

Verschwindet ein Fehler oder seine Ursache nach einer bestimmten Zeit wieder, wird dieses Ereignis ebenfalls über die EMCY-Nachricht signalisiert. Die zuletzt aufgetretenen Fehler werden im Objektverzeichnis (Objekt 1003_{16}) abgelegt und können über einen SDO-Zugriff ausgelesen werden (→ *CANx_SDO_READ*, → Seite 164). Zusätzlich spiegelt sich die aktuelle Fehlersituation im Error-Register (Objekt 1001_{16}) wider.

Man unterscheidet folgende Fehler:

8046

Kommunikationsfehler

- Der CAN-Controller signalisiert CAN-Fehler.
  (Das gehäufte Auftreten ist ein Indiz für physikalische Probleme. Diese Fehler können einen erheblichen Einfluss auf das Übertragungsverhalten und damit auf den Datendurchsatz eines Netzwerks haben.)
- Life-Guarding- oder Heartbeat-Fehler

Anwendungsfehler

- Kurzschluss oder Leiterbruch
- Temperatur zu hoch

Aufbau einer Fehlernachricht

8047

Eine Fehlernachricht (EMCY Message) hat folgenden Aufbau:

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
EMCY-Fehlercode, wie im Objekt 1003_{16} eingetragen		Objekt 1001_{16}	Herstellerspezifische Informationen				

Identifier

8048

Der Identifier für die Fehlernachricht besteht aus der Summe folgender Elemente:

EMCY-Default-Identifier 128 (80_{16})
+
Node-ID

© ifm electronic gmbh

EMCY-Fehlercode

8049

Er gibt detailliert Auskunft darüber, welcher Fehler aufgetreten ist. Eine Liste möglicher Fehlercodes ist bereits im Kommunikationsprofil definiert. Fehlercodes, die nur für eine bestimmte Geräteklasse gültig sind, werden im jeweiligen Geräteprofil dieser Geräteklasse festgelegt.

Objekt 0x1003 (Error Field)

8050

Das Objekt 1003_{16} stellt den Fehlerspeicher eines Gerätes dar. Die Subindizes enthalten die zuletzt aufgetretenen Fehler, die ein Fehler-Telegramm ausgelöst haben.

Tritt ein neuer Fehler auf, dann wird sein EMCY-Fehlercode immer im Subindex 1_{16} gespeichert. Alle anderen, älteren Fehler werden im Fehlerspeicher um einen Platz nach hinten geschoben, also der Subindex um 1 erhöht. Falls alle unterstützten Subindizes belegt sind, wird der älteste Fehler gelöscht. Der Subindex 0_{16} wird auf die Anzahl der gespeicherten Fehler erhöht. Nachdem alle Fehler behoben sind, wird in das Fehlerfeld des Subindex 1_{16} der Wert "0" geschrieben.

Um den Fehlerspeicher zu löschen, kann der Subindex 0_{16} mit dem Wert "0" beschrieben werden. Andere Werte dürfen nicht eingetragen werden.

Gerätefehler signalisieren

1880

Wie beschrieben, werden EMCY-Nachrichten versendet, wenn Fehler in einem Gerät auftreten. Im Unterschied zu frei programmierbaren Geräten, werden beispielsweise von dezentralen Ein-/Ausgangsmodulen (z.B. CompactModule CR2033) Fehlermeldungen automatisch verschickt. Entsprechende Fehler-Codes → jeweiliges Gerätehandbuch.

Die programmierbaren Geräte erzeugen nur dann automatisch eine EMCY-Nachricht (z.B. Kurzschluss an einem Ausgang), wenn *CANx_MASTER_EMCY_HANDLER* (→ Seite 142) oder *CANx_SLAVE_EMCY_HANDLER* (→ Seite 155) in das Applikations-Programm eingebunden wird.

Übersicht der automatisch verschickten EMCY-Fehlercodes für alle mit CoDeSys programmierbaren **ecomatmobile**-Geräte → Kapitel *Übersicht CANopen Error-Codes*, → Seite 175.

Sollen zusätzlich noch applikations-spezifische Fehler durch das Applikations-Programm verschickt werden, werden *CANx_MASTER_SEND_EMERGENCY* (→ Seite 144) oder *CANx_SLAVE_SEND_EMERGENCY* (→ Seite 157) eingesetzt.

© ifm electronic gmbh

7.6.3 Übersicht CANopen Error-Codes

8545

Error Code (hex)	Meaning / Bedeutung
00xx	Reset or no Error (Fehler rücksetzen / kein Fehler)
10xx	Generic Error (allgemeiner Fehler)
20xx	Current (Stromfehler)
21xx	Current, device input side (Stromfehler, eingangsseitig)
22xx	Current inside the device (Stromfehler im Geräteinnern)
23xx	Current, device output side (Stromfehler, ausgangsseitig)
30xx	Voltage (Spannungsfehler)
31xx	Mains Voltage
32xx	Voltage inside the device (Spannungsfehler im Geräteinnern)
33xx	Output Voltage (Spannungsfehler, ausgangsseitig)
40xx	Temperature (Temperaturfehler)
41xx	Ambient Temperature (Umgebungstemperaturfehler)
42xx	Device Temperature (Gerätetemperaturfehler)
50xx	Device Hardware (Geräte-Hardware-Fehler)
60xx	Device Software (Geräte-Software-Fehler)
61xx	Internal Software (Firmware-Fehler)
62xx	User Software (Applications-Software)
63xx	Data Set (Daten-/Parameterfehler)
70xx	Additional Modules (zusätzliche Module)
80xx	Monitoring (Überwachung)
81xx	Communication (Kommunikation)
8110	CAN Overrun-objects lost (CAN Überlauf-Datenverlust)
8120	CAN in Error Passiv Mode (CAN im Modus "fehlerpassiv")
8130	Life Guard Error or Heartbeat Error (Guarding-Fehler oder Heartbeat-Fehler)
8140	Recovered from Bus off (Bus-Off zurückgesetzt)
8150	Transmit COB-ID collision (Senden "Kollision des COB-ID")
82xx	Protocol Error (Protokollfehler)
8210	PDO not procedded due to length error (PDO nicht verarbeitet, fehlerhafte Längenangabe)
8220	PDO length exceeded (PDO Längenfehler, ausgangsseitig)
90xx	External Error (Externer Fehler)
F0xx	Additional Functions (zusätzliche Funktionen)
FFxx	Device specific (gerätespezifisch)

© ifm electronic gmbh

Objekt 0x1001 (Error-Register)

8547

Dieses Objekt spiegelt den allgemeinen Fehlerzustand eines CANopen-Gerätes wider. Das Gerät ist dann als fehlerfrei anzusehen, wenn das Objekt 1001_{16} keinen Fehler mehr signalisiert.

Bit	Meaning (Bedeutung)
0	Generic Error (allgemeiner Fehler)
1	Current (Stromfehler)
2	Voltage (Spannungsfehler)
3	Temperature (Temperaturfehler)
4	Communication Error (Kommunikationsfehler)
5	Device Profile specific (Geräteprofil spezifisch)
6	Reserved – always 0 (reserviert – immer 0)
7	manufacturer specific (herstellerspezifisch)

Für eine Fehlermeldung können mehrere Bits im Error-Register gleichzeitig gesetzt sein.

Beispiel: CR2033, Meldung "Leitungsbruch" an Kanal 2 (→ Installationsanleitung des Geräts):

COB-ID	DLC	Byte 0	Byte 1	Byte	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4
80_{16} + Node-ID		**00**	**FF**	**81**	**10**	**00**	00	00	00

Error-Code = $FF00_{16}$

Error-Register = 81_{16} = $1000\ 0001_2$, besteht also aus folgenden Fehlern:
- Generic Error (allgemeiner Fehler)
- manufacturer specific (herstellerspezifisch)

Betroffener Kanal = 0010_{16} = $0000\ 0000\ 0001\ 0000_2$ = Kanal 2

© ifm electronic gmbh

Herstellerspezifische Informationen

8548

Hier kann ein Gerätehersteller zusätzliche Fehlerinformationen mitteilen. Das Format ist dabei frei wählbar.

Beispiel:

In einem Gerät treten zwei Fehler auf und werden über den Bus gemeldet:

- Kurzschluss der Ausgänge:
  Fehlercode 2300_{16},
  im Objekt 1001_{16} wird der Wert 03_{16} ($0000\ 0011_2$) eingetragen
  (allg. Fehler und Stromfehler)

- CAN-Überlauf:
  Fehlercode 8110_{16},
  im Objekt 1001_{16} wird der Wert 13_{16} ($0001\ 0011_2$) eingetragen
  (allg. Fehler, Stromfehler und Kommunikationsfehler)

>> CAN-Überlauf bearbeitet:
Fehlercode 0000_{16},
im Objekt 1001_{16} wird der Wert 03_{16} ($0000\ 0011_2$) eingetragen
(allg. Fehler, Stromfehler, Kommunikationsfehler zurückgesetzt.)

Nur aus dieser Information kann man entnehmen, dass der Kommunikationsfehler nicht mehr anliegt.

Übersicht CANopen EMCY-Codes (CR105n, CR106n)

3713

alle Angaben (hex) für 1. CAN-Schnittstelle

EMCY-Code Objekt 1003_{16}		Objekt 1001_{16}	Hersteller-spezifische Informationen					
Byte 0	1	2	3	4	5	6	7	Beschreibung
00	80	11						CAN1 Monitoring SYNC-Error (nur Slave)
10	81	11						CAN1 Empfangspuffer Überlauf
11	81	11						CAN1 Sendepuffer Überlauf
30	81	11						CAN1 Guard-/Heartbeat-Error (nur Slave)

© ifm electronic gmbh

8 Ein-/Ausgangs-Funktionen

Inhalt:

Eingangswerte verarbeiten 178
Analoge Werte anpassen 180

1590

Hier zeigen wir Ihnen Funktionen zum Lesen und Bearbeiten der Signale an den Ein- und Ausgängen.

8.1 Eingangswerte verarbeiten

Inhalt:

TOGGLE 179

1602

Hier zeigen wir Ihnen Funktionen zum Lesen und Verarbeiten der analogen oder digitalen Signale am Geräte-Eingang.

HINWEIS

Die in der Steuerungskonfiguration von CoDeSys erscheinenden Rohwerte kommen direkt aus dem ADU. Sie sind noch nicht korrigiert!

Deshalb können in der Steuerungskonfiguration bei gleichen Geräten unterschiedliche Rohwerte erscheinen.

Erst durch die **ifm**-FBs (z.B. INPUT, INPUT_ANALOG) findet eine Fehlerkorrektur und Normierung statt. Die FBs liefern den korrigierten Wert.

© ifm electronic gmbh

8.1.1 TOGGLE

3194

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek:	Für folgende Geräte verfügbar:
`ifm_PDM_UTIL_Vxxyyzz.LIB`	- PDM360: CR1050, CR1051 - PDM360compact: CR1052, CR1053, CR1055, CR1056
`ifm_PDMng_UTIL_Vxxyyzz.LIB`	- PDM360NG: CR108n
`ifm_PDMsmart_UTIL_Vxxyyzz.LIB`	- PDM360smart: CR1070, CR1071

Symbol in CoDeSys:

Beschreibung

3304

TOGGLE ermöglicht das Setzen und Rücksetzen einer boolschen Variablen mit nur einem Eingangs-Bit.

Die erste steigende Flanke am Eingang IN setzt den Ausgang OUT auf 'TRUE'.
Die nächste steigende Flanke setzt den Ausgang wieder zurück auf 'FALSE'.
usw.

Parameter der Eingänge

3305

Parameter	Datentyp	Beschreibung
IN	BOOL	Flanke FALSE ⇨ TRUE: Setzen / Rücksetzen des Ausgangs

Parameter der Ausgänge

3306

Parameter	Datentyp	Beschreibung
OUT	BOOL	1. Flanke an IN ⇨ TRUE 2. Flanke an IN ⇨ FALSE 3. Flanke an IN ⇨ TRUE ...

© ifm electronic gmbh

8.2 Analoge Werte anpassen

Inhalt:

NORM_DINT 181
NORM_REAL 183

1603

Wenn die Werte analoger Eingänge oder die Ergebnisse von analogen Funktionen angepasst werden müssen, helfen Ihnen die folgenden Funktionsblöcke.

© ifm electronic gmbh

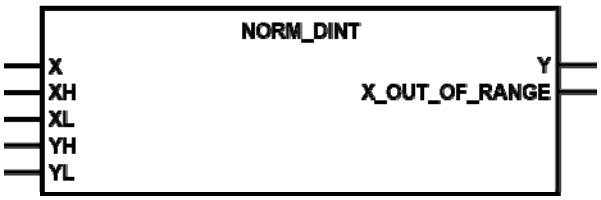
8.2.1 NORM_DINT

3200

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek:	Für folgende Geräte verfügbar:
`ifm_PDM_UTIL_Vxxyyzz.LIB`	- PDM360: CR1050, CR1051 - PDM360compact: CR1052, CR1053, CR1055, CR1056
`ifm_PDMng_UTIL_Vxxyyzz.LIB`	- PDM360NG: CR108n
`ifm_PDMsmart_UTIL_Vxxyyzz.LIB`	- PDM360smart: CR1070, CR1071

Symbol in CoDeSys:



Beschreibung

3307

NORM_DINT normiert einen Wert innerhalb festgelegter Grenzen auf einen Wert mit neuen Grenzen.

Der FB normiert einen Wert vom Typ DINT, der innerhalb der Grenzen XH und XL liegt, auf einen Ausgangswert innerhalb der Grenzen YH und YL. Der FB wird z.B. bei der Erzeugung von PWM-Werten aus analogen Eingangsgrößen genutzt.

HINWEIS

Der Wert für X muss sich im definierten Eingangsbereich zwischen XL und XH befinden (es findet keine interne Plausibilitätsprüfung des Wertes statt). Außerhalb dieses Wertebereiches wird der Ausgang X_OUT_OF_RANGE gesetzt.

Bedingt durch die Rundungsfehler können Abweichungen beim normierten Wert um 1 auftreten.

Werden die Grenzen (XH/XL oder YH/YL) invertiert angegeben, erfolgt auch die Normierung invertiert.

Parameter der Eingänge

3308

Parameter	Datentyp	Beschreibung
X	DINT	aktueller Eingangswert
XH	DINT	obere Grenze des Eingangswertebereich
XL	DINT	untere Grenze des Eingangswertebereich
YH	DINT	obere Grenze des Ausgangswertebereich
YL	DINT	untere Grenze des Ausgangswertebereich

© ifm electronic gmbh

Parameter der Ausgänge

3309

Parameter	Datentyp	Beschreibung
Y	DINT	normierter Wert
X_OUT_OF_RANGE	BOOL	Eingangswert X ist außerhalb des definierten Wertebereichs XL/XH

© ifm electronic gmbh

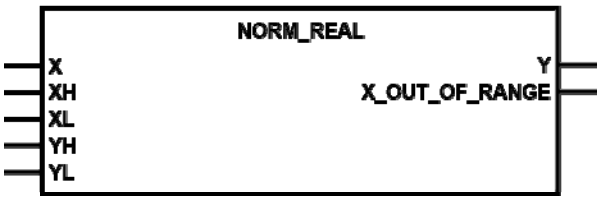
8.2.2 NORM_REAL

3202

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek:	Für folgende Geräte verfügbar:
`ifm_PDM_UTIL_Vxxyyzz.LIB`	- PDM360: CR1050, CR1051 - PDM360compact: CR1052, CR1053, CR1055, CR1056
`ifm_PDMng_UTIL_Vxxyyzz.LIB`	- PDM360NG: CR108n
`ifm_PDMsmart_UTIL_Vxxyyzz.LIB`	- PDM360smart: CR1070, CR1071

Symbol in CoDeSys:



Beschreibung

3310

NORM_REAL normiert einen Wert innerhalb festgelegter Grenzen auf einen Wert mit neuen Grenzen.

Der FB normiert einen Wert vom Typ REAL, der innerhalb der Grenzen XH und XL liegt, auf einen Ausgangswert innerhalb der Grenzen YH und YL. Der FB wird z.B. bei der Erzeugung von PWM-Werten aus analogen Eingangsgrößen genutzt.

HINWEIS

Der Wert für X muss sich im definierten Eingangsbereich zwischen XL und XH befinden (es findet keine interne Plausibilitätsprüfung des Wertes statt). Außerhalb dieses Wertebereiches wird der Ausgang X_OUT_OF_RANGE gesetzt.

Bedingt durch die Rundungsfehler können Abweichungen beim normierten Wert um 1 auftreten.

Werden die Grenzen (XH/XL oder YH/YL) invertiert angegeben, erfolgt auch die Normierung invertiert.

Parameter der Eingänge

3311

Parameter	Datentyp	Beschreibung
X	REAL	aktueller Eingangswert
XH	REAL	obere Grenze des Eingangswertebereich
XL	REAL	untere Grenze des Eingangswertebereich
YH	REAL	obere Grenze des Ausgangswertebereich
YL	REAL	untere Grenze des Ausgangswertebereich

© ifm electronic gmbh

Parameter der Ausgänge

3312

Parameter	Datentyp	Beschreibung
Y	REAL	normierter Wert
X_OUT_OF_RANGE	BOOL	Eingangswert X ist außerhalb des definierten Wertebereichs XL/XH

© ifm electronic gmbh

9 Daten verwalten

Inhalt:

Systemzeit lesen / schreiben 185
Datenzugriff und Datenprüfung 190
Dateien verwalten 194

8606

Hier zeigen wir Ihnen Funktionen, mit denen Sie Daten im Gerät lesen und verarbeiten können.

9.1 Systemzeit lesen / schreiben

Inhalt:

PDM_READ_RTC 186
PDM_SET_RTC 188

1601

Mit folgenden Bausteinen der **ifm electronic gmbh** können Sie die kontinuierlich laufende Systemzeit des Controllers lesen und im Applikations-Programm auswerten oder bei Bedarf ändern.

© ifm electronic gmbh

9.1.1 PDM_READ_RTC

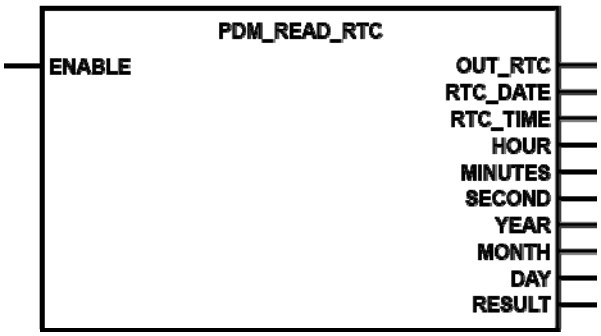
3172

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_CRnnnn_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1053, CR1056

Symbol in CoDeSys:



Beschreibung

3277

PDM_READ_RTC liest die aktuelle Uhrzeit aus der Echtzeituhr (Hardware-RTC) des PDM aus.

Das PDM ist mit einer batteriegestützen Echtzeituhr (Real time clock = RTC) ausgerüstet. Die Echtzeituhr muss einmalig auf Datum und Uhrzeit eingestellt werden, danach läuft sie auch ohne Versorgungsspannung weiter. Das Einstellen der Uhr kann von Ihnen oder dem Anwender über folgende Möglichkeiten erfolgen:

- im Setup-Programm → Kapitel *Echtzeituhr (RTC) einstellen*, → Seite 20 oder
- im Applikations-Programm mit *PDM_SET_RTC* → Seite 188.

Zur Laufzeit des Applikations-Programms können Sie die Uhrzeit und das Datum ausgelesen und weiterverarbeiten. Mögliche Einsatzzwecke sind z.B.:

- Zeitstempel zusammen mit Daten von Ereignissen (z.B. Störungen) speichern.
- Maschinenlaufzeiten berechnen zur Planung von Wartungsintervallen.
- Sperre gegen unbefugtes Benutzen zu bestimmten Zeiten.

HINWEIS

Dieser Baustein verursacht eine relativ hohe Systembelastung.

► Deshalb den Baustein nur kurzzeitig bei Bedarf freischalten.

► Besser: Zum Auslesen der Systemzeit folgende 3S-CoDeSys-Bausteine verwenden:

- SysRtcGetTime aus der Bibliothek `SysLibRtc.Lib` oder
- CurTimeEx aus der Bibliothek `SysLibTime.Lib`.

Diese Bausteine synchronisieren sich beim Gerätestart mit der Hardware-RTC.
→ CoDeSys-Onlinehilfe unter den jeweiligen Funktionen.

© ifm electronic gmbh

SysRtcGetTime und CurTimeEx werden NICHT snychronisiert, wenn die Hardware-RTC während der Laufzeit mit *PDM_SET_RTC* (→ Seite 188) gestellt wird.

Parameter der Eingänge

3278

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv

Parameter der Ausgänge

3279

Parameter	Datentyp	Beschreibung
OUT_RTC	STRING (19)	Aktueller Wert der RTC als Jahr / Monat / Tag / Stunde / Minute / Sekunde, zusammengefasst in einem String
RTC_date	STRING (80)	Aktuelles Datum der RTC als Jahr / Monat / Tag, zusammengefasst in einem String
RTC_time	STRING (80)	Aktuelle Uhrzeit der RTC als Stunde / Minute / Sekunde, zusammengefasst in einem String
HOUR	BYTE	Stunden
MINUTES	BYTE	Minuten
SECOND	BYTE	Sekunden
YEAR	WORD	Jahre
MONTH	BYTE	Monate
DAY	BYTE	Tage
RESULT	DINT	Hilfsausgang, kann bei Bedarf zur Fehlerdiagnose genutzt werden: RESULT = -1 ⇨ Lesefehler

© ifm electronic gmbh

9.1.2 PDM_SET_RTC

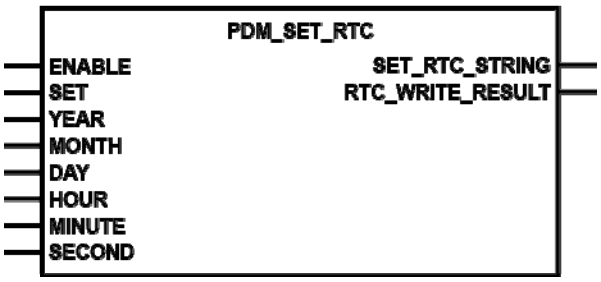
3174

Baustein-Typ = Programm (PRG)

Enthalten in Bibliothek: `ifm_CRnnnn_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1053, CR1056

Symbol in CoDeSys:



Beschreibung

3280

PDM_SET_RTC stellt die Uhrzeit der Echtzeituhr (Hardware-RTC) des PDM auf einen bestimmbaren Wert ein.

Das PDM ist mit einer batteriegestützen Echtzeituhr (Real time clock = RTC) ausgerüstet. Die Echtzeituhr muss einmalig auf Datum und Uhrzeit eingestellt werden, danach läuft sie auch ohne Versorgungsspannung weiter. Das Einstellen der Uhr kann von Ihnen oder dem Anwender über folgende Möglichkeiten erfolgen:

- im Setup-Programm → Kapitel *Echtzeituhr (RTC) einstellen*, → Seite 20 oder
- im Applikations-Programm mit PDM_SET_RTC.

Zur Laufzeit des Applikations-Programms können Sie die Uhrzeit und das Datum auslesen und weiterverarbeiten. Mögliche Einsatzzwecke sind z.B.:

- Zeitstempel zusammen mit Daten von Ereignissen (z.B. Störungen) speichern.
- Maschinenlaufzeiten berechnen zur Planung von Wartungsintervallen.
- Sperre gegen unbefugtes Benutzen zu bestimmten Zeiten.

© ifm electronic gmbh

Parameter der Eingänge

3281

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv
SET	BOOL	TRUE: Werte RTC werden mit den neuen Werten gesetzt FALSE: Werte RTC bleiben unverändert
YEAR	WORD	Jahre
MONTH	BYTE	Monate
DAY	BYTE	Tage
HOUR	BYTE	Stunden
MINUTES	BYTE	Minuten
SECOND	BYTE	Sekunden

Parameter der Ausgänge

3282

Parameter	Datentyp	Beschreibung
SET_RTC_STRING	STRING (80)	Geschriebener Wert der RTC als Jahr / Monat / Tag / Stunde / Minute / Sekunde, zusammengefasst in einem String
RTC_WRTIE_RESULT	DINT	Hilfsausgang, kann bei Bedarf zur Fehlerdiagnose genutzt werden: RTC_WRTIE_RESULT = -1 ⇨ Schreibfehler

© ifm electronic gmbh

9.2 Datenzugriff und Datenprüfung

Inhalt:

DEVICE_KERNEL_VERSION2 191
DEVICE_RUNTIME_VERSION 192
DEVICE_TS..... 193

1598

Die Bausteine in diesem Kapitel steuern den Datenzugriff und ermöglichen ein Prüfen der Daten.

© ifm electronic gmbh

9.2.1 DEVICE_KERNEL_VERSION2

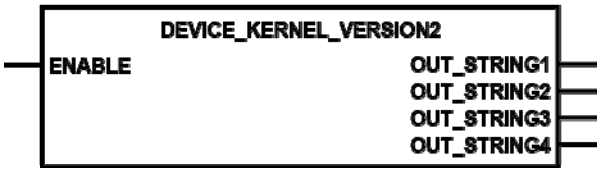
3211

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_UTIL_Vxxyyzz.Lib`

Für folgende Geräte verfügbar:
- PDM360compact: CR1052, CR1053

Symbol in CoDeSys:



Beschreibung

3316

DEVICE_KERNEL_VERSION2 liest die Linux Kernel-Version beim PDM360compact.

Mit dem FB kann direkt die Version des Linux-Kernels ausgelesen werden. Die von dem FB erzeugten Strings werden direkt passend für die Anzeige auf dem Display formatiert.

Parameter der Eingänge

3314

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE (steigende Flanke): Baustein wird ausgeführt (nur 1 Zyklus lang) FALSE: Baustein wird nicht ausgeführt Baustein-Ein- und Ausgänge sind nicht aktiv

Parameter der Ausgänge

3317

Parameter	Datentyp	Beschreibung
OUT_STRING1	STRING(40)	Linux-Version, 1. Teil
OUT_STRING2	STRING(40)	Linux-Version, 2. Teil
OUT_STRING3	STRING(40)	Linux-Version, 3. Teil
OUT_STRING4	STRING(40)	Linux-Version, 4. Teil

© ifm electronic gmbh

9.2.2 DEVICE_RUNTIME_VERSION

3205

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_UTIL_Vxxyyzz.Lib`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Symbol in CoDeSys:

Beschreibung

3318

DEVICE_RUNTIME_VERSION liest die Version des Laufzeitsystems.

Parameter der Eingänge

3314

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE (steigende Flanke): Baustein wird ausgeführt (nur 1 Zyklus lang) FALSE: Baustein wird nicht ausgeführt Baustein-Ein- und Ausgänge sind nicht aktiv

Parameter der Ausgänge

3319

Parameter	Datentyp	Beschreibung
OUT_STRING	STRING(5)	Version des Laufzeitsystems

© ifm electronic gmbh

9.2.3 DEVICE_TS

3208

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_UTIL_Vxxyyzz.Lib`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Symbol in CoDeSys:

Beschreibung

3320

DEVICE_TS liest den Teile-Status beim PDM360 und PDM360compact.

Der FB zeigt bei alten Projekten den Teile-Status des Geräts an. Der Teile-Status von aktuellen Laufzeitsystemen wird nicht mehr angezeigt. Die Versionsverwaltung erfolgt dann nur noch über die Version des Laufzeitsystems → *DEVICE_RUNTIME_VERSION*, → Seite 192.

Parameter der Eingänge

3314

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE (steigende Flanke): Baustein wird ausgeführt (nur 1 Zyklus lang) FALSE: Baustein wird nicht ausgeführt Baustein-Ein- und Ausgänge sind nicht aktiv

Parameter der Ausgänge

3321

Parameter	Datentyp	Beschreibung
OUT_STRING	STRING(1)	Teile-Status des Geräts

© ifm electronic gmbh

9.3 Dateien verwalten

Inhalt:

Dateifunktionen 194
Asynchrone Dateifunktionen 233

8611

Hier zeigen wir Ihnen Funktionen, mit denen Sie Dateien und ihre Inhalte verwalten können.

9.3.1 Dateifunktionen

Inhalt:

Allgemeine Dateifunktionen..... 196
Dateifunktionen für Textdateien 201
Dateifunktionen für CSV-Dateien 205
Dateifunktionen für Parameterdateien 209
Dateifunktionen für die Alarmverwaltung 215
Hilfsfunktionen für Arrays 221
Weitere Hilfsfunktionen 229

3357

Da das PDM auf dem Linux-Betriebssystem aufbaut, stehen Ihnen für die Sicherung und Verarbeitung von Daten und Parametern vielfältige und mächtige Dateifunktionen zur Verfügung. Ähnlich wie man es von einem PC kennt, können Dateien auf die verschiedenen Laufwerke (PCMCIA-Speicherkarte oder RAM- oder FLASH-Disk oder USB-Laufwerk) geschrieben und von diesen gelesen werden. Es können auch Datensätze im CSV-Format erzeugt und gelesen werden. Diese Daten können anschließend z.B. in MS-Excel verarbeitet werden.

Die Dateifunktionen des PDM lassen sich vielfältig einsetzen und beliebig kombinieren.

HINWEIS

Beim PDM360 handelt es sich um ein Embedded-System, das in vielen Funktionen vergleichbar wie ein PC arbeitet, aber nicht die gleichen Leistungsreserven bereitstellt.

► Daher unbedingt auf die richtige und maßvolle Anwendung der Funktionen achten! Andernfalls können sich negative Auswirkungen auf das Laufverhalten und die Systemstabilität ergeben.

Die Bibliothek enthält folgende Funktionen:

- **File_Functions**
  - Allgemeine Dateifunktionen
    - *DELETE_FILE*, → Seite 197
    - *OPEN_PCMCIA*, → Seite 198
    - *OPEN_PCMCIA_VFAT*, → Seite 199
    - *SYNC*, → Seite 200
  - Funktionen für Textdateien
    - *READ_TEXTFILE*, → Seite 202
    - *READ_TEXTLINE*, → Seite 203
  - Funktionen für CSV-Dateien
    - *READ_CSV_8BYTE*, → Seite 206
    - *WRITE_CSV_8BYTE*, → Seite 207

© ifm electronic gmbh

- Funktionen für Parameterdateien
  - *READ_PARAMETER_WORD*, → Seite 210
  - *WRITE_PARAMETER_WORD*, → Seite 212
  - *READ_SINGLE_PARAMETER*, → Seite 213
  - *WRITE_SINGLE_PARAMETER*, → Seite 214
- Funktionen für die Alarmverwaltung
  - *READ_ALARM*, → Seite 216
  - *STORE_ALARM*, → Seite 218

- **Help_Functions**
  - Funktionen für Arrays
    - *ARRAY_TO_WORD*, → Seite 222
    - *ARRAY_TO_DWORD*, → Seite 223
    - *WORD_TO_ARRAY*, → Seite 224
    - *DWORD_TO_ARRAY*, → Seite 225
    - *BYTE_ARRAY_TO_CSV*, → Seite 226
    - *BYTE_ARRAY_TO_STRING*, → Seite 227
    - *READBUFFER_TO_STRING*, → Seite 228
  - Weitere Funktionen
    - *PDM_FILE_VERSION*, → Seite 230
    - *DUMP_FILESYSTEM*, → Seite 231
    - *LOG_TIMESTAMP*, → Seite 232

HINWEIS

Die in der Bibliothek enthaltenen Hilfsfunktionen werden intern benötigt und sollten nicht für eigene Projekte genutzt werden. Die Funktionen können sich ggf. bei Updates der Bibliothek ändern.

Die Bibliothek wurde in IEC erstellt und ist nicht geschützt. So können die Funktionen bei Bedarf von erfahrenen Programmierern erweitert oder angepasst werden.

© ifm electronic gmbh

Allgemeine Dateifunktionen

Inhalt:

DELETE_FILE 197
OPEN_PCMCIA 198
OPEN_PCMCIA_VFAT 199
SYNC 200

3428

Allgemeine Funktionen zur Dateiverwaltung:

- DELETE_FILE
  Für folgende Geräte verfügbar:
  - PDM360: CR1050, CR1051
  - PDM360compact: CR1052, CR1053, CR1055, CR1056
  - PDM360NG: CR108n
- OPEN_PCMCIA
  Für folgende Geräte verfügbar:
  - PDM360: CR1050, CR1051
- OPEN_PCMCIA_VFAT
  Für folgende Geräte verfügbar:
  - PDM360: CR1050, CR1051
- SYNC
  Für folgende Geräte verfügbar:
  - PDM360: CR1050, CR1051
  - PDM360compact: CR1052, CR1053, CR1055, CR1056
  - PDM360NG: CR108n

© ifm electronic gmbh

DELETE_FILE

3392

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:

Beschreibung

3495

DELETE_FILE löscht die angegebene Datei vom internen Flash-Speicher oder von der externen PCMCIA-Karte.

Parameter der Eingänge

3496

Parameter	Datentyp	Beschreibung
DEL	BOOL	Flanke FALSE ⇨ TRUE: Baustein wird ausgeführt (nur 1 Zyklus lang) FALSE: Baustein wird nicht ausgeführt
FILENAME	STRING(80)	Pfad und Dateiname der zu löschenden Datei, z.B.: `'/home/project/dummy.txt'`

© ifm electronic gmbh

OPEN_PCMCIA

3395

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051

Symbol in CoDeSys:

Beschreibung

3497

OPEN_PCMCIA öffnet das FAT-Dateisystem der gesteckten PCMCIA-Karte (bindet die PCMCIA-Karte ein) als Verzeichnis `/pcmcia`.

Das Linux-System muss einen neu gesteckten Wechseldatenträger in sein Dateisystem einbinden, um die Inhalte lesen und bearbeiten zu können. Das System erledigt dies automatisch mit dem Kommando:
`'/dev/hda1 on /pcmcia type msdos (rw)$N'`

Parameter der Eingänge

3498

Parameter	Datentyp	Beschreibung
INIT	BOOL	TRUE (im 1. Zyklus): Baustein wird ausgeführt FALSE: im weiteren Programmablauf

Parameter der Ausgänge

3499

Parameter	Datentyp	Beschreibung
PCMCIA_STATE	STRING(80)	nur informell: Mount-Status
OPEN_ERROR	BOOL	TRUE: Mount-Status-Datei ist nicht lesbar: > Einbinden der Karte ist fehlgeschlagen FALSE: Mount-Status-Datei ist lesbar
PCMCIA_READY	BOOL	TRUE: PCMCIA-Karte fertig eingebunden: > Dateien im Pfad `/pcmcia` können gelesen und bearbeitet werden FALSE: PCMCIA-Karte nicht eingebunden

© ifm electronic gmbh

OPEN_PCMCIA_VFAT

3469

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051 (ab Geräte-Version 4.12)

Symbol in CoDeSys:

Beschreibung

3502

OPEN_PCMCIA öffnet das VFAT-Dateisystem der gesteckten PCMCIA-Karte (bindet die PCMCIA-Karte ein) als Verzeichnis `/pcmcia`.

Das Linux-System muss einen neu gesteckten Wechseldatenträger in sein Dateisystem einbinden, um die Inhalte lesen und bearbeiten zu können. Dies erfolgt mit dem FB per Kommando: `'/dev/hda1 on /pcmcia type vfat (rw)$N'`.

Parameter der Eingänge

3498

Parameter	Datentyp	Beschreibung
INIT	BOOL	TRUE (im 1. Zyklus): Baustein wird ausgeführt FALSE: im weiteren Programmablauf

Parameter der Ausgänge

3499

Parameter	Datentyp	Beschreibung
PCMCIA_STATE	STRING(80)	nur informell: Mount-Status
OPEN_ERROR	BOOL	TRUE: Mount-Status-Datei ist nicht lesbar: > Einbinden der Karte ist fehlgeschlagen FALSE: Mount-Status-Datei ist lesbar
PCMCIA_READY	BOOL	TRUE: PCMCIA-Karte fertig eingebunden: > Dateien im Pfad `/pcmcia` können gelesen und bearbeitet werden FALSE: PCMCIA-Karte nicht eingebunden

© ifm electronic gmbh

SYNC

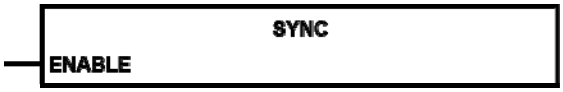
3475

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:



Beschreibung

3504

SYNC synchronisiert den Flash-Speicher (oder den PCMCIA-Speicher) mit dem Arbeitsspeicher RAM.

Dieser FB sollte von Zeit zu Zeit und als letzter vor dem Herunterfahren des PDM verwendet werden.

Parameter der Eingänge

3505

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE (steigende Flanke): Baustein wird ausgeführt (nur 1 Zyklus lang) FALSE: Baustein wird nicht ausgeführt Baustein-Ein- und Ausgänge sind nicht aktiv

© ifm electronic gmbh

Dateifunktionen für Textdateien

Inhalt:

READ_TEXTFILE 202
READ_TEXTLINE 203

3429

Textdateien lassen sich mit folgenden FBs bearbeiten:

- READ_TEXTFILE
- READ_TEXTLINE

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

© ifm electronic gmbh

READ_TEXTFILE

3398

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:

Beschreibung

3506

READ_TEXTFILE liest den Inhalt einer angegebenen Textdatei. Diese Textdatei kann z.B. einen Sprachsatz oder Alarmtexte enthalten.

Die Datei darf max. 500 Zeichen (Bytes) enthalten, verteilt auf max. 25 Zeilen von je max. 65 Zeichen. Werden mehr Zeichen benötigt, sollte der Inhalt auf mehrere Dateien verteilt werden.
Für große Textdateien mit vielen Zeilen besser *READ_TEXTLINE* (→ Seite 203) verwenden.

Der erfahrene Programmierer kann die Anzahl der Zeichen und Zeilen bei Bedarf ändern. Dabei mit Vorsicht vorgehen und die Zykluszeit des Gerätes prüfen.

Parameter der Eingänge

3507

Parameter	Datentyp	Beschreibung
READ	BOOL	TRUE (steigende Flanke): Baustein wird ausgeführt (nur 1 Zyklus lang) FALSE: im weiteren Programmablauf
PATH_FILE	STRING(80)	Pfad und Name der Textdatei, z.B.: `'/home/project/english.txt'`

Parameter der Ausgänge

3508

Parameter	Datentyp	Beschreibung
TEXT_LINES	ARRAY[1..25] OF STRING	Gelesene Zeilen aus der Textdatei
FILEDES_	DWORD	Datei-Beschreibung: Wert < 1 = Fehler
READ_RESULT	DINT	Anzahl der gelesenen Zeichen

© ifm electronic gmbh

READ_TEXTLINE

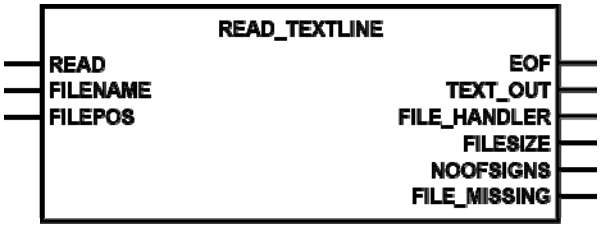
3401

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:



Beschreibung

3509

READ_TEXTLINE liest aus einer angegebenen Textdatei eine Alarm- oder Textzeile.

Der FB ermöglicht direkten Zugriff auf eine bestimmte Textzeile in der Datei. Auch große Textdateien mit vielen Zeilen können schnell eingelesen werden. Die Systembelastung ist gering.
Für kleine Textdateien besser *READ_TEXTFILE* (→ Seite 202) verwenden.

Alle Zeilen in dieser Datei müssen exakt 39 Zeichen lang sein, abgeschlossen mit 'CR'. Nicht benötigte Schreibstellen der Zeile mit z.B. '_' oder Leerzeichen füllen.

Beispiele:

```
Alarmtext 1_____________________________
Out of order ..........................
```

Parameter der Eingänge

3510

Parameter	Datentyp	Beschreibung
READ	BOOL	TRUE (steigende Flanke): Baustein wird ausgeführt (nur 1 Zyklus lang) FALSE: im weiteren Programmablauf
FILENAME	STRING(80)	Pfad und Name der Textdatei, z.B.: `'/home/project/alarm.txt'`
FILE_POS	WORD	Nummer der Textzeile

© ifm electronic gmbh

Parameter der Ausgänge

3511

Parameter	Datentyp	Beschreibung
EOF	BOOL	TRUE: Dateiende erreicht FASLE: Dateiende noch nicht erreicht
TEXT_OUT	STRING(39)	Gelesene Zeile aus der Textdatei
FILE_HANDLER	DWORD	Datei-Beschreibung: Wert < 1 = Fehler
FILE_SIZE	DWORD	Dateigröße [Bytes]
NoOfSigns	DWORD	Anzahl der gelesenen Zeichen
FILE_MISSING	BOOL	Datei nicht gefunden: - falscher Dateiname - falscher Pfad - Datei nicht im PDM gespeichert

© ifm electronic gmbh

Dateifunktionen für CSV-Dateien

Inhalt:

READ_CSV_8BYTE 206
WRITE_CSV_8BYTE 207

3430

CSV = (Comma-Separated Values) Komma-separierte Werte

Eine CSV-Datei ist eine Textdatei, die in jeder Zeile durch Komma getrennte Elemente enthält. Jede Zeile beschreibt einen Datensatz. Mehrere Datensätze sind durch Zeilenschaltung getrennt. Auf diese einfache Weise ergibt sich eine Tabelle (Matrix, Array), die in einem Tabellenkalkulations-Programm (z.B. MS-Excel, StarCalc, ...) weiterverarbeitet werden kann.

Beispiel:

Date,Time,a-p,Error
20080812,845,am,17
20080812,256,pm,3
20080813,923,am,12

Aus den vorstehenden Daten der CSV-Datei ergibt sich folgende Tabelle:

Date	Time	a-p	Error
2008-08-12	08:45	am	17
2008-08-12	02:56	pm	3
2008-08-13	09:23	am	12

Die Bibliothek stellt folgende FBs für CSV-Dateien zur Verfügung:

- READ_CSV_8BYTE
- WRITE_CSV_8BYTE

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

© ifm electronic gmbh

READ_CSV_8BYTE

3404

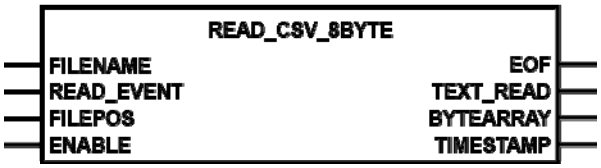
Baustein-Typ = Funktionsblock (FB)

Funktionsfähig ab CoDeSys Version 2.3.6.1. Nicht für ältere Projekte verwenden!

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:



Beschreibung

3512

READ_CSV_8BYTE liest aus einer angegebenen Datei eine Byte-Matrix im CSV-Format.

Die Datei muss mit *WRITE_CSV_8BYTE* (→ Seite 207) geschrieben worden sein.
Jeder Datensatz (jede Zeile) muss aus 8 Bytes bestehen.

Der FB liefert den gewählten Datensatz und den zugehörigen Zeitstempel.

Parameter der Eingänge

3513

Parameter	Datentyp	Beschreibung
FILENAME	STRING(80)	Pfad und Name der CSV-Datei z.B.: `'/home/project/data.csv'`
READ_EVENT	BOOL	Flanke FALSE ⇨ TRUE: Datensatz wird gelesen (nur 1 Zyklus lang) FALSE: im weiteren Programmablauf
FILE_POS	WORD	Nummer des zu lesenden Datensatzes (der Zeile)
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv

Parameter der Ausgänge

3514

Parameter	Datentyp	Beschreibung
EOF	BOOL	TRUE: Dateiende erreicht FASLE: Dateiende noch nicht erreicht
TEXT_READ	STRING(53)	Gelesener Datensatz aus der Datei, bestehend aus: `Zeitstempel,Byte1,Byte2,...,Byte8`
BYTEARRAY	ARRAY[1...8] OF BYTE	Gelesener Datensatz aus der Datei als Matrix von 8 Bytes
TIMESTAMP	STRING(80)	Zeitstempel des gelesenen Datensatzes

© ifm electronic gmbh

WRITE_CSV_8BYTE

3407

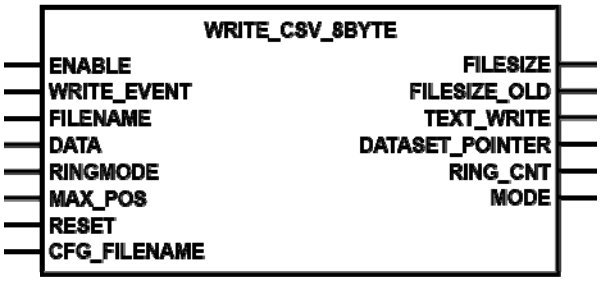
Baustein-Typ = Funktionsblock (FB)

Funktionsfähig ab CoDeSys Version 2.3.6.1. Nicht für ältere Projekte verwenden!

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:



Beschreibung

3515

WRITE_CSV_8BYTE schreibt eine Matrix von 8 Bytes als Datensatz in eine angegebene CSV-Datei.

Der FB arbeitet im Auto-Inkrement-Modus. Dabei wird automatisch bei jedem Speichervorgang der Zeiger um 1 erhöht (= inkrementiert).

Datei-Aufbau im Linear-Modus vs. Ring-Modus

Der Speicher kann mit 2 verschiedenen Methoden beschrieben werden:

a) im Ring-Modus:
Die Daten werden bis zu einer in MAX_POS angegebenen Dateigrenze (Anzahl von Datensätzen) in den Speicher geschrieben. Danach beginnt der Schreibvorgang automatisch wieder an der ersten Speicherstelle. Die bisherigen Datensätze werden dabei überschrieben. Der Speichervorgang wird lückenlos fortgeführt.

b) im Linear-Modus:
Die Daten werden bis zu einer in MAX_POS angegebenen Dateigrenze (Anzahl von Datensätzen) in den Speicher geschrieben. Damit ist der Schreibvorgang automatisch beendet. Es wird kein Datensatz überschrieben.

Prinzip:

Datensatz-Nummer	Datensatz-Inhalt
1	Timestamp,Byte1,Byte2,...,ByteMax
2	Timestamp,Byte1,Byte2,...,ByteMax
3	Timestamp,Byte1,Byte2,...,ByteMax
:	:
MAX_POS	Timestamp,Byte1,Byte2,...,ByteMax

© ifm electronic gmbh

Parameter der Eingänge

3516

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv
WRITE_EVENT	BOOL	Flanke FALSE ⇨ TRUE: Datensatz wird geschrieben (nur 1 Zyklus lang) FALSE: im weiteren Programmablauf
FILENAME	STRING(80)	Pfad und Name der CSV-Datei, z.B.: `'/home/project/data.csv'`
DATA	ARRAY[1...8] OF BYTE	Datensatz aus 8 Bytes
RINGMODE	BOOL	TRUE: Ring-Modus FALSE: Linear-Modus
MAX_POS	DWORD	Max. Nummer des zu schreibenden Datensatzes (der Zeile)
RESET	BOOL	Datei und Datensatz-Zeiger löschen
CFG_FILENAME	STRING(80)	Für Ring-Modus: enthält den letzten Datensatz-Zeiger, Pfad und Name der Zeiger-Datei, z.B.: `'/home/project/pointer.csv'`

Parameter der Ausgänge

3517

Parameter	Datentyp	Beschreibung
FILESIZE	DINT	aktuelle Dateigröße [Bytes]
FILESIZE_OLD	DINT	bisherige Dateigröße [Bytes] (informativ)
TEXT_WRITE	STRING(55)	Geschriebener Datensatz
DATASET_POINTER	DINT	Nummer des aktuellen Datensatzes
RING_CNT	DWORD	Zähler für RINGMODE
MODE	STRING(80)	Aktuelle Betriebsart: - 'ring' - 'linear'

© ifm electronic gmbh

Dateifunktionen für Parameterdateien

Inhalt:

READ_PARAMETER_WORD 210
WRITE_PARAMETER_WORD 212
READ_SINGLE_PARAMETER 213
WRITE_SINGLE_PARAMETER 214

3431

Für Parameterdateien stellt die Bibliothek folgende FBs bereit:

- READ_PARAMETER_WORD
- WRITE_PARAMETER_WORD
- READ_SINGLE_PARAMETER
- WRITE_SINGLE_PARAMETER

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

© ifm electronic gmbh

READ_PARAMETER_WORD

3410

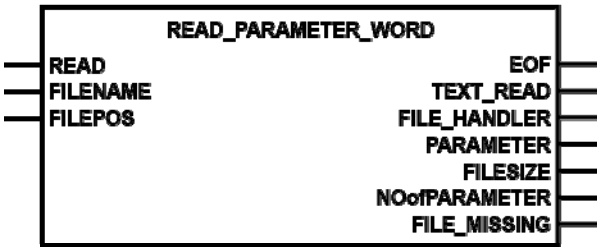
Baustein-Typ = Funktionsblock (FB)

Funktionsfähig ab CoDeSys Version 2.3.6.1. Nicht für ältere Projekte verwenden!

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:



Beschreibung

3518

READ_PARAMETER_WORD liest aus der angegebenen Textdatei Datensätze vom Typ WORD.

Die Daten müssen in der Textdatei rechtsbündig gespeichert sein, z.B.:

Wert	zu speichern als
1	00001
123	00123
12345	12345

Parameter der Eingänge

3519

Parameter	Datentyp	Beschreibung
READ	BOOL	Flanke FALSE ⇨ TRUE: Datensatz wird gelesen (nur 1 Zyklus lang) FALSE: im weiteren Programmablauf
FILENAME	STRING(80)	Pfad und Name der Datei, z.B.: `'/home/project/para.txt'`
FILEPOS	WORD	Nummer (1...n) des zu lesenden Datensatzes (der Zeile)

© ifm electronic gmbh

Parameter der Ausgänge

3520

Parameter	Datentyp	Beschreibung
EOF	BOOL	TRUE: Dateiende erreicht FASLE: Dateiende noch nicht erreicht
TEXT_READ	STRING(6)	Gelesene Zeile aus der Textdatei
FILE_HANDLER	DWORD	Datei-Beschreibung: Wert < 1 = Fehler
PARAMETER	WORD	Gelesener Parameter aus der Textdatei
FILESIZE	DWORD	Dateigröße [Bytes]
NoOfParameter	DWORD	Nummer des gelesenen Parameters
FILE_MISSING	BOOL	Datei nicht gefunden: - falscher Dateiname - falscher Pfad - Datei nicht im PDM gespeichert

© ifm electronic gmbh

WRITE_PARAMETER_WORD

3413

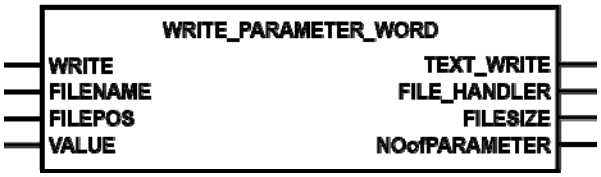
Baustein-Typ = Funktionsblock (FB)

Funktionsfähig ab CoDeSys Version 2.3.6.1. Nicht für ältere Projekte verwenden!

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:



Beschreibung

3521

WRITE_PARAMETER_WORD schreibt in die angegebene Textdatei Datensätze vom Typ WORD.

Die Daten werden in der Textdatei rechtsbündig gespeichert, z.B.:

Wert	gespeichert als
1	00001
123	00123
12345	12345

Parameter der Eingänge

3522

Parameter	Datentyp	Beschreibung
WRITE	BOOL	Flanke FALSE ⇨ TRUE: Parameter wird geschrieben (nur 1 Zyklus lang) FALSE: im weiteren Programmablauf
FILENAME	STRING(80)	Pfad und Name der Datei, z.B.: `'/home/project/para.txt'`
FILEPOS	WORD	Nummer (0...n) des zu schreibenden Datensatzes (der Zeile)
VALUE	WORD	Wert, der in den Datensatz geschrieben werden soll

Parameter der Ausgänge

3523

Parameter	Datentyp	Beschreibung
TEXT_WRITE	STRING(6)	Geschriebener Parameter als Datensatz
FILE_HANDLER	DWORD	Datei-Beschreibung: Wert < 1 = Fehler
FILESIZE	DWORD	Dateigröße [Bytes]
NoOfParameter	DWORD	Nummer des geschriebenen Parameters

© ifm electronic gmbh

READ_SINGLE_PARAMETER

3416

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:

Beschreibung

3524

READ_SINGLE_PARAMETER liest aus der angegebenen Textdatei einen einzelnen Datensatz vom Typ WORD. Die Datei besteht aus nur diesem einen Datensatz.

Der Wert muss in der Textdatei rechtsbündig gespeichert sein, z.B.:

Wert	zu speichern als
1	00001
123	00123
12345	12345

Parameter der Eingänge

3525

Parameter	Datentyp	Beschreibung
READ	BOOL	Flanke FALSE ⇨ TRUE: Datensatz wird gelesen (nur 1 Zyklus lang) FALSE: im weiteren Programmablauf
FILENAME	STRING(80)	Pfad und Name der Datei, z.B.: `'/home/project/para.txt'`

Parameter der Ausgänge

3526

Parameter	Datentyp	Beschreibung
FILE_HANDLER	DWORD	Datei-Beschreibung: Wert < 1 = Fehler
PARAMETER	WORD	Gelesener Parameter aus der Textdatei
FILE_MISSING	BOOL	Datei nicht gefunden: - falscher Dateiname - falscher Pfad - Datei nicht im PDM gespeichert

© ifm electronic gmbh

WRITE_SINGLE_PARAMETER

3419

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:

Beschreibung

3527

WRITE_SINGLE_PARAMETER schreibt in die angegebene Textdatei einen Datensatz vom Typ WORD. Die Datei besteht aus nur diesem einen Datensatz.

Der Wert wird in der Textdatei als Datensatz rechtsbündig gespeichert, z.B.:

Wert	gespeichert als
1	00001
123	00123
12345	12345

Parameter der Eingänge

3528

Parameter	Datentyp	Beschreibung
WRITE	BOOL	Flanke FALSE ⇨ TRUE: Datensatz wird geschrieben (nur 1 Zyklus lang) FALSE: im weiteren Programmablauf
FILENAME	STRING(80)	Pfad und Name der Datei z.B.: `'/home/project/para.txt'`
VALUE	WORD	Wert, der in den Datensatz geschrieben werden soll

Parameter der Ausgänge

3529

Parameter	Datentyp	Beschreibung
FILE_HANDLER	DWORD	Datei-Beschreibung: Wert < 1 = Fehler

© ifm electronic gmbh

Dateifunktionen für die Alarmverwaltung

Inhalt:

READ_ALARM 216
STORE_ALARM 218

3432

Hierfür bietet die Bibliothek folgende FBs:

- READ_ALARM
- STORE_ALARM

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

© ifm electronic gmbh

READ_ALARM

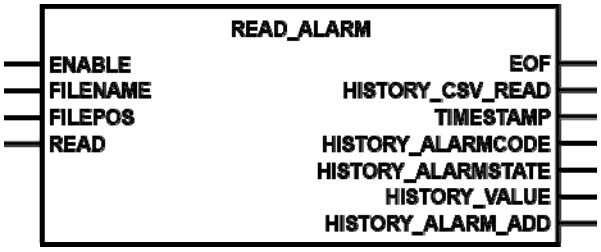
3422

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:



Beschreibung

3530

READ_ALARM liest Alarm-Meldungen aus einer CSV-Datei.
→ *STORE_ALARM*, → Seite 218

Struktur der Alarmmeldungen

3616

Struktur der Alarmmeldung im CSV-Datensatz

Name	Datentyp	Beschreibung
Timestamp	Byte	Weitere Information zum Alarm.
Alarmcode	Byte 0	Alarmcode, auch Index in der ALARMSTATE-Matrix.
	Byte 1	
Alarmstate	Bool	Status der Alarmmeldung: TRUE: Alarm kommt, FALSE: Alarm geht.
Alarmaddition	Byte	Weitere Information zum Alarm.
Alarmvalue	Byte 0	Alarm-Wert (Wert der Größe, welche die aktuelle Alarmmeldung ausgelöst hat).
	Byte 1	
	Byte 2	
	Byte 3	

Dateiaufbau

Pointer	Dateiaufbau im Linearmode
1	Timestamp, Alarmcode (Byte 0), Alarmcode (Byte 1), Alarmstate,..., Alarmvalue (Byte 3)
2	Timestamp, Alarmcode (Byte 0), Alarmcode (Byte 1), Alarmstate,..., Alarmvalue (Byte 3)
3	Timestamp, Alarmcode (Byte 0), Alarmcode (Byte 1), Alarmstate,..., Alarmvalue (Byte 3)
:	:
n	Timestamp, Alarmcode (Byte 0), Alarmcode (Byte 1), Alarmstate,..., Alarmvalue (Byte 3)

© ifm electronic gmbh

Parameter der Eingänge

3531

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv
FILENAME	STRING(80)	Pfad und Name der CSV-Datei z.B.: `'/home/project/alarm.csv'`
FILE_POS	WORD	Nummer des zu lesenden Datensatzes in der Alarm-Geschichte
READ	BOOL	Flanke FALSE ⇨ TRUE: Datensatz wird gelesen (nur 1 Zyklus lang) FALSE: im weiteren Programmablauf

Parameter der Ausgänge

3532

Parameter	Datentyp	Beschreibung
EOF	BOOL	TRUE: Dateiende erreicht FASLE: Dateiende noch nicht erreicht
HISTORY_csv_READ	STRING(80)	Gelesener Datensatz aus der Datei, bestehend aus: `Zeitstempel, Alarmcode LSB+MSB, Alarmstatus EIN/AUS (1/0), weitere Information, History-Wert LSB+MSB.`
TIMESTAMP	STRING(80)	Zeitstempel des gelesenen Datensatzes
History_Alarmcode	WORD	Alarmcode, kann in der Applikation z.B. mit einer Text-Matrix verknüpft werden
History_Alarmstate	BOOL	TRUE: Alarmstatus = EIN FALSE: Alarmstatus = AUS
History_Value	DWORD	History-Wert: Wann kam und wann ging der Alarm?
History_Alarm_Add	BYTE	Weitere Information: Das ist der eigentliche Wert zum Alarmzeitpunkt (z.B. Temperatur) oder der Wert zum Zeitpunkt, als der Alarm wieder "gegangen" ist.

© ifm electronic gmbh

STORE_ALARM

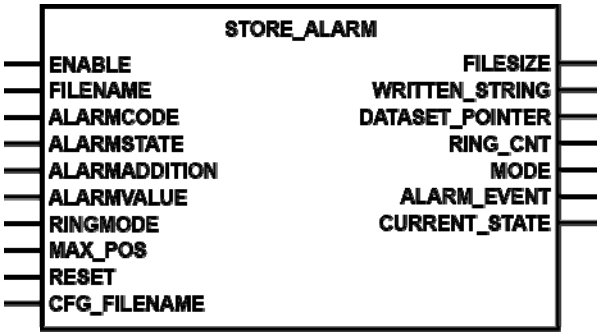
3425

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:



Beschreibung

3533

STORE_ALARM schreibt Alarm-Meldungen zusammen mit einem Zeitstempel in eine CSV-Datei.

Pro Alarmdatei können 255 unterschiedliche Alarme verwaltet werden.

► Sie sollten daher zwei Arrays der Länge 255 anlegen. Das erste Array enthält die Klartextmeldungen der Alarme:

Index	Text (STRING)
1	Übertemperatur
2	Öldruck zu hoch
3	Überdrehzahl
:	:
255	Kurzschluss Ventil 5

► Im zweiten Array wird der Alarmstatus der einzelnen Alarme gespeichert:

Index	Status (BOOL)
1	TRUE
2	FALSE
3	FALSE
:	:
255	TRUE

Über den gleichlautenden Index, der in der Regel auch der Alarmcode sein sollte, kann eine Verknüpfung zwischen den Arrays hergestellt und der Klartext auf dem Display ausgegeben werden.

© ifm electronic gmbh

Tritt nun ein Alarmereignis auf, wird durch das Applikations-Programm der Alarmstatus der Alarmmeldung von FALSE auf TRUE gesetzt und der Alarmcode (d.h. der Array-Index des geänderten Array-Elements) an STORE_ALARM und ggf. die Zusatzinformation übergeben. STORE_ALARM erkennt die Veränderung des Alarmstatus und speichert die Meldung automatisch mit einem Zeitstempel in der Alarmdatei ab. Solange sich der Alarmstatus dieser oder einer anderen Meldung nicht ändert, wird kein neuer Eintrag in der Alarmdatei gemacht.

Die Alarmdatei kann bei Bedarf direkt im Gerät mit *READ_ALARM* (→ Seite 216) verarbeitet werden oder per FTP an einen PC übertragen und dort direkt mit z.B. Excel bearbeitet werden.

Datei-Aufbau im Linear-Modus vs. Ring-Modus

Der Speicher kann mit 2 verschiedenen Methoden beschrieben werden:

a) im Ring-Modus:
Die Daten werden bis zu einer in MAX_POS angegebenen Dateigrenze (Anzahl von Datensätzen) in den Speicher geschrieben. Danach beginnt der Schreibvorgang automatisch wieder an der ersten Speicherstelle. Die bisherigen Datensätze werden dabei überschrieben. Der Speichervorgang wird lückenlos fortgeführt.

b) im Linear-Modus:
Die Daten werden bis zu einer in MAX_POS angegebenen Dateigrenze (Anzahl von Datensätzen) in den Speicher geschrieben. Damit ist der Schreibvorgang automatisch beendet. Es wird kein Datensatz überschrieben.

Prinzip:

Datensatz-Nummer	Datensatz-Inhalt
1	Timestamp,Byte1,Byte2,...,ByteMax
2	Timestamp,Byte1,Byte2,...,ByteMax
3	Timestamp,Byte1,Byte2,...,ByteMax
:	:
MAX_POS	Timestamp,Byte1,Byte2,...,ByteMax

Struktur der Alarmmeldungen

3616

Struktur der Alarmmeldung im CSV-Datensatz

Name	Datentyp	Beschreibung
Timestamp	Byte	Weitere Information zum Alarm.
Alarmcode	Byte 0	Alarmcode, auch Index in der ALARMSTATE-Matrix.
	Byte 1	
Alarmstate	Bool	Status der Alarmmeldung: TRUE: Alarm kommt, FALSE: Alarm geht.
Alarmaddition	Byte	Weitere Information zum Alarm.
Alarmvalue	Byte 0	Alarm-Wert (Wert der Größe, welche die aktuelle Alarmmeldung ausgelöst hat).
	Byte 1	
	Byte 2	
	Byte 3	

© ifm electronic gmbh

Dateiaufbau

Pointer	Dateiaufbau im Linearmode
1	Timestamp, Alarmcode (Byte 0), Alarmcode (Byte 1), Alarmstate,..., Alarmvalue (Byte 3)
2	Timestamp, Alarmcode (Byte 0), Alarmcode (Byte 1), Alarmstate,..., Alarmvalue (Byte 3)
3	Timestamp, Alarmcode (Byte 0), Alarmcode (Byte 1), Alarmstate,..., Alarmvalue (Byte 3)
:	:
n	Timestamp, Alarmcode (Byte 0), Alarmcode (Byte 1), Alarmstate,..., Alarmvalue (Byte 3)

Parameter der Eingänge

3534

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv
FILENAME	STRING(80)	Pfad und Name der CSV-Datei z.B.: `'/home/project/alarm.csv'`
ALARMCODE	WORD	Alarmcode, auch Index in der ALARMSTATE-Matrix
ALARMSTATE	ARRAY[1...255] OF BOOL	Enthält den Status aller definierten Alarmmeldungen: TRUE: Alarm kommt FALSE: Alarm geht
ALARMADDITION	BYTE	Weitere Information zum Alarm
ALARMVALUE	DWORD	Alarm-Wert (Wert der Größe, welche die aktuelle Alarmmeldung ausgelöst hat)
RINGMODE	BOOL	TRUE: Im Ring-Modus speichern FALSE: Im Linear-Modus speichern
MAX_POS	DWORD	Max. Anzahl der Datensätze
RESET	BOOL	Datei-Inhalt (Alarmliste) und Datensatz-Zeiger löschen
CFG_FILENAME	STRING(80)	Für Ring-Modus: enthält den letzten Datensatz-Zeiger, Pfad und Name der Zeiger-Datei z.B.: `'/home/project/pointer.txt'`

Parameter der Ausgänge

3535

Parameter	Datentyp	Beschreibung
FILESIZE	DWORD	aktuelle Dateigröße [Bytes]
WRITTEN_STRING	STRING(80)	Geschriebener Datensatz
DATASET_POINTER	DWORD	Nummer des nächsten Datensatzes
RING_CNT	DWORD	Zähler für Anzahl der Überläufe des Ringspeichers
MODE	STRING(80)	Aktuelle Betriebsart: - 'ring' - 'linear'
ALARM_EVENT	BOOL	Alarm-Fall, TRUE nur 1 Zyklus lang
CURRENT_STATE	BOOL	Status des aktuellen Alarms

© ifm electronic gmbh

Hilfsfunktionen für Arrays

Inhalt:

ARRAY_TO_WORD 222
ARRAY_TO_DWORD 223
WORD_TO_ARRAY 224
DWORD_TO_ARRAY 225
BYTE_ARRAY_TO_CSV 226
BYTE_ARRAY_TO_STRING 227
READBUFFER_TO_STRING 228

3433

Hierfür bietet die Bibliothek folgende Bausteine:

- ARRAY_TO_WORD
- ARRAY_TO_DWORD
- WORD_TO_ARRAY
- DWORD_TO_ARRAY
- BYTE_ARRAY_TO_CSV
- BYTE_ARRAY_TO_STRING
- READBUFFER_TO_STRING

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

© ifm electronic gmbh

ARRAY_TO_WORD

3434

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:

Beschreibung

3536

ARRAY_TO_WORD ist eine Hilfsfunktion für andere Bausteine.

Der FB fügt den Inhalt von 2 Bytes zu einem Wort zusammen nach folgendem Muster:

Wort	WORD															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Byte	BYTE 1								BYTE 0							

Parameter der Eingänge

3537

Parameter	Datentyp	Beschreibung
BYTE_0	BYTE	LSB im Ausgangs-Wort
BYTE_1	BYTE	HSB im Ausgangs-Wort

Parameter der Ausgänge

3538

Parameter	Datentyp	Beschreibung
WORD_OUT	WORD	Ausgangs-Wort

© ifm electronic gmbh

ARRAY_TO_DWORD

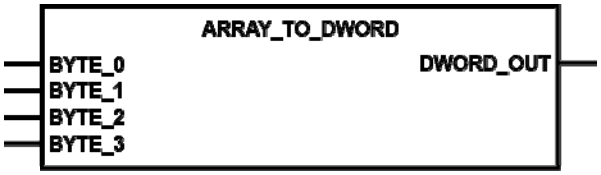
3437

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:



Beschreibung

3539

ARRAY_TO_DWORD ist eine Hilfsfunktion für andere Bausteine.

Der FB fügt den Inhalt von 4 Bytes zu einem Doppelwort zusammen nach folgendem Muster:

Wort	DWORD															
Bit	30 31	28 29	26 27	24 25	22 23	20 21	18 19	16 17	14 15	12 13	10 11	8 9	6 7	4 5	2 3	0 1
Byte	BYTE 3				BYTE 2				BYTE 1				BYTE 0			

Parameter der Eingänge

3540

Parameter	Datentyp	Beschreibung
BYTE_0	BYTE	LSB im niederen Ausgangs-Wort
BYTE_1	BYTE	HSB im niederen Ausgangs-Wort
BYTE_2	BYTE	LSB im hohen Ausgangs-Wort
BYTE_3	BYTE	HSB im hohen Ausgangs-Wort

Parameter der Ausgänge

3541

Parameter	Datentyp	Beschreibung
DWORD_OUT	DWORD	Ausgangs-Doppelwort

© ifm electronic gmbh

WORD_TO_ARRAY

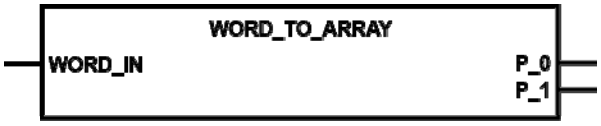
3440

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:



Beschreibung

3542

WORD_TO_ARRAY ist eine Hilfsfunktion für andere Bausteine.

Der FB trennt ein Eingangs-Wort in 2 getrennte Bytes auf nach folgendem Muster:

Wort	WORD															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Byte	BYTE 1								BYTE 0							

Parameter der Eingänge

3543

Parameter	Datentyp	Beschreibung
WORD_IN	WORD	Eingangs-Wort

Parameter der Ausgänge

3544

Parameter	Datentyp	Beschreibung
P_0	BYTE	LSB im Eingangs-Wort
P_1	BYTE	HSB im Eingangs-Wort

© ifm electronic gmbh

DWORD_TO_ARRAY

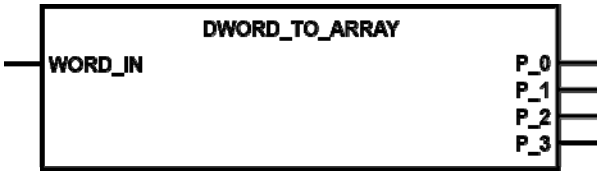
3443

Baustein-Typ = Funktion (FUN) vom Typ BYTE

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:



Beschreibung

3545

DWORD_TO_ARRAY ist eine Hilfsfunktion für andere Bausteine.

Der FB trennt den Inhalt eines Doppelworts in 4 Bytes auf nach folgendem Muster:

Wort	DWORD															
Bit	30 31	28 29	26 27	24 25	22 23	20 21	18 19	16 17	14 15	12 13	10 11	8 9	6 7	4 5	2 3	0 1
Byte	BYTE 3				BYTE 2				BYTE 1				BYTE 0			

Parameter der Eingänge

3546

Parameter	Datentyp	Beschreibung
DWORD_OUT	DWORD	Eingangs-Doppelwort

Parameter der Ausgänge

3547

Parameter	Datentyp	Beschreibung
P_0	BYTE	LSB im niederen Eingangs-Wort
P_1	BYTE	HSB im niederen Eingangs-Wort
P_2	BYTE	LSB im hohen Eingangs-Wort
P_3	BYTE	HSB im hohen Eingangs-Wort

© ifm electronic gmbh

BYTE_ARRAY_TO_CSV

3446

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:

Beschreibung

3548

BYTE_ARRAY_TO_CSV ist eine Hilfsfunktion für andere Bausteine.

Der FB fügt den Inhalt von 8 Bytes zu einem String zusammen nach folgendem Muster:

`H-Byte7,L-Byte7,H-Byte6,L-Byte6,H-Byte5,L-Byte5,H-Byte4,L-Byte4,H-Byte3,L-Byte3,H-Byte2,L-Byte2,H-Byte1,L-Byte1,H-Byte0,L-Byte0`

H-Byte = High-Byte
L-Byte = Low-Byte

Parameter der Eingänge

3549

Parameter	Datentyp	Beschreibung
DATA	ARRAY[1...8] OF BYTE	Matrix aus 1...8 Bytes

Parameter der Ausgänge

3550

Parameter	Datentyp	Beschreibung

CSV_OUT	STRING(32)	Ausgabewert in der Form:
		`H-Byte7,L-Byte7,H-Byte6,L-Byte6,`
		`H-Byte5,L-Byte5,H-Byte4,L-Byte4,`
		`H-Byte3,L-Byte3,H-Byte2,L-Byte2,`
		`H-Byte1,L-Byte1,H-Byte0,L-Byte0`

H-Byte = High-Byte,
L-Byte = Low-Byte.

Jedes Halb-Byte ist im String durch Komma separiert.

© ifm electronic gmbh

BYTE_ARRAY_TO_STRING

3449

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:

Beschreibung

3551

BYTE_ARRAY_TO_STRING ist eine Hilfsfunktion für andere Bausteine.

Der FB fügt den Inhalt von bis zu 65 Bytes zu einem String zusammen nach folgendem Muster:

`Byte65,Byte64,Byte63,...,Byte1,Byte0`

Parameter der Eingänge

3552

Parameter	Datentyp	Beschreibung
BYTE_IN	ARRAY[1...65] OF BYTE	Matrix aus 1...65 Bytes

Parameter der Ausgänge

3553

Parameter	Datentyp	Beschreibung
OUT_STRING	STRING(80)	Ausgabewert in der Form: `Byte65,Byte64,Byte63,...,Byte1,Byte0`

© ifm electronic gmbh

READBUFFER_TO_STRING

3452

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:

Beschreibung

3554

READBUFFER_TO_STRING ist eine Hilfsfunktion für andere Bausteine.

Der FB fügt den Inhalt einer Datei (max. 500 Bytes) zu einer Anzahl von Strings zusammen.

Parameter der Eingänge

3555

Parameter	Datentyp	Beschreibung
BYTEARRAY	ARRAY[1...500] OF BYTE	Matrix aus 1...500 Bytes als gelesener Inhalt einer Datei (Lesepuffer)

Parameter der Ausgänge

3556

Parameter	Datentyp	Beschreibung
TEXT_LINE	ARRAY[1...25] OF STRING(80)	Matrix der gelesenen Textzeilen
NO	BYTE	Anzahl der gefundenen Textzeilen
FORMAT_ERROR	BOOL	TRUE: gelesenes Datenformat fehlerhaft FALSE: gelesenes Datenformat okay

© ifm electronic gmbh

Weitere Hilfsfunktionen

Inhalt:

PDM_FILE_VERSION 230
DUMP_FILESYSTEM 231
LOG_TIMESTAMP 232

3455

Hierfür bietet die Bibliothek folgende Bausteine:

- PDM_FILE_VERSION
- DUMP_FILESYSTEM
- LOG_TIMESTAMP

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

© ifm electronic gmbh

PDM_FILE_VERSION

3456

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:

Beschreibung

3357

PDM_FILE_VERSION ist eine Hilfsfunktion.

Dieser FB liefert die Version dieser Bibliothek.

Parameter der Ausgänge

3558

Parameter	Datentyp	Beschreibung
VERSION	STRING(80)	Version des Dateisystems

© ifm electronic gmbh

DUMP_FILESYSTEM

3459

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:

Beschreibung

3559

DUMP_FILESYSTEM ist eine Hilfsfunktion für andere Bausteine.

Dieser FB liefert Informationen über das Dateisystems entsprechend dem Linux-Kommando `df`.

Parameter der Eingänge

3560

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv
READ_DF	BOOL	Flanke FALSE ⇨ TRUE: Daten werden gelesen (1 Zyklus) FALSE: im weiteren Programmablauf

Parameter der Ausgänge

3561

Parameter	Datentyp	Beschreibung
FILE_INFO_TEXT	ARRAY[1..25] OF STRING(80)	Datei-Informationen

© ifm electronic gmbh

LOG_TIMESTAMP

3462

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_FILE_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1053, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:

Beschreibung

3562

LOG_TIMESTAMP ist eine Hilfsfunktion für andere Bausteine.

Dieser FB wandelt die CurTimeEx in ein CSV-kompatibles Format.

Parameter der Eingänge

3563

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv

Parameter der Ausgänge

3564

Parameter	Datentyp	Beschreibung
TIMESTAMP_STRING	STRING(19)	aktuelle Zeit im CSV-kompatiblen Format

© ifm electronic gmbh

9.3.2 Asynchrone Dateifunktionen

Inhalt:

FILE_AVAIL 235
FILE_MAX 236
FILE_READ 237
FILE_WRITE 239

3658

Die asynchronen Dateifunktionen aus der Bibliothek `ifm_SysLibFile_Async_Vxxyyzz.Lib` arbeiten im Hintergrund, ohne den IEC-Task zu unterbrechen – im Gegensatz zu den 3S-Standard-Funktionen. Folglich sind die asynchronen **ifm**-Funktionen für den SPS-Zyklus völlig unkritisch.

Im folgenden Bild 1 sehen Sie, wie IEC-Task A von Task B unterbrochen wird. Task A muss dazu seinen Zwischenstand speichern und nach Ablauf von Task B wieder einlesen, bevor TASK A fortgesetzt und beendet werden kann:

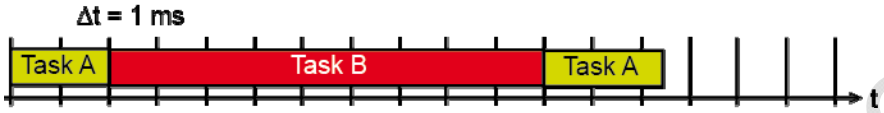


Bild 1: Synchrone 3S-Funktion (Task B) unterbricht einen IEC-Task A

Bild 2 dagegen zeigt, dass der IEC-Task A unbehindert weiterläuft, während der schnelle, asynchrone Task B im Hintergrund arbeitet. Der gesamte SPS-Zyklus ist somit deutlich kürzer:

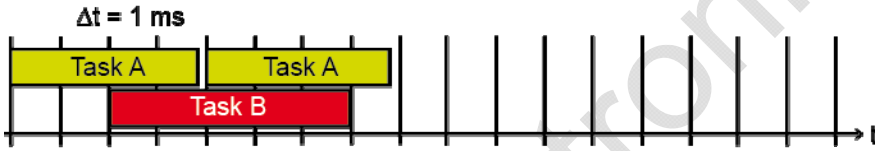


Bild 2: Asynchrone ifm-Funktion (Task B) arbeitet parallel zu mehreren IEC-Tasks A

Asynchrone Funktionen erlauben problemloses Schreiben und Lesen von großen Datenblöcken. Sie können aber auch kleine Datensätze unterschiedlicher Länge in eine Datei schreiben, wobei jedem Datensatz zur Identifizierung eine Ende-Markierung angehängt wird.

Asynchrone Funktionen erlauben relativ schnelles Aufzeichnen (Loggen) von Daten, indem diese Daten zunächst im RAM (IEC-Datenbereich) zwischengespeichert werden. Ist der dafür vorgesehene Bereich voll oder die Aufzeichnung beendet, schreibt *FILE_WRITE* (→ Seite 239) den kompletten Block in die angegebene Datei.

Info

Das Aufzeichen der Daten wird beim Speichern des Datenblocks nicht unterbrochen, wenn Sie mit 2 getrennten IEC-Datenbereichen arbeiten, die Sie abwechselnd mit Daten füllen.

Je nach Zykluszeit des Applikations-Programms sind Zykluszeiten bis herunter zu 10 ms möglich.

© ifm electronic gmbh

Beispiel:

Für eine Applikation schreibt eine Funktion in dichter Folge Datensätze in einen Datenbereich.

- Sobald dieser Datenbereich voll ist, laufen die nachfolgenden Datensätze in den zweiten Datenbereich. Gleichzeitig kopiert FILE_WRITE die Datensätze aus dem ersten Datenbereich als `Block 1` in den Flash-Speicher.
- Sobald auch der zweite Datenbereich gefüllt ist, laufen die nachfolgenden Datensätze in den ersten Datenbereich. Gleichzeitig kopiert FILE_WRITE die Datensätze aus dem zweiten Datenbereich als `Block 2` in den Flash-Speicher.
- Sobald auch der erste Datenbereich erneut gefüllt ist, laufen die nachfolgenden Datensätze in den zweiten Datenbereich. Gleichzeitig kopiert FILE_WRITE die Datensätze aus dem ersten Datenbereich als `Block 3` in den Flash-Speicher.
- usw.

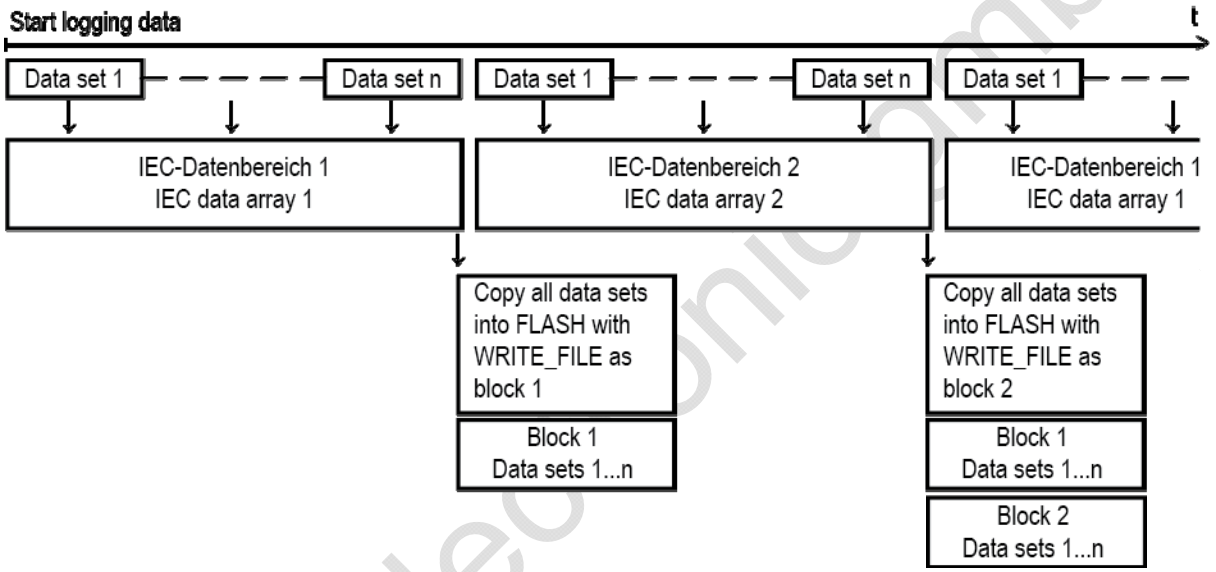


Bild 3: Beispiel für asynchrones Speichern während der kontinuierlichen Datenaufzeichnung

Die Bibliothek enthält folgende Funktionen:

- *FILE_AVAIL* (→ Seite 235) liefert Anzahl der Dateien, die noch geöffnet werden dürfen.
- *FILE_MAX* (→ Seite 236) liefert maximale Anzahl der Dateien, die gleichzeitig geöffnet sein dürfen.
- *FILE_READ* (→ Seite 237) liest asynchron eine Datei.
- *FILE_WRITE* (→ Seite 239) speichert asynchron eine Datei.

© ifm electronic gmbh

FILE_AVAIL

3663

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_SysLibFile_Async_Vxxyyzz.Lib`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Symbol in CoDeSys:

Beschreibung

3680

FILE_AVAIL liefert Anzahl der Dateien, die noch geöffnet werden dürfen.

Parameter der Ausgänge

3681

Parameter	Datentyp	Beschreibung
FILE_AVAIL	WORD	Anzahl der Dateien, die noch geöffnet werden dürfen

© ifm electronic gmbh

FILE_MAX

3667

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_SysLibFile_Async_Vxxyyzz.Lib`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Symbol in CoDeSys:

Beschreibung

3682

FILE_MAX liefert die maximale Anzahl der Dateien, die gleichzeitig geöffnet sein dürfen.

Parameter der Ausgänge

3683

Parameter	Datentyp	Beschreibung
FILE_MAX	WORD	maximale Anzahl der Dateien, die gleichzeitig geöffnet sein dürfen

© ifm electronic gmbh

FILE_READ

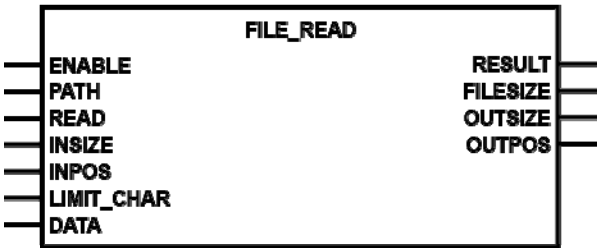
3670

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_SysLibFile_Async_Vxxyyzz.Lib`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Symbol in CoDeSys:



Beschreibung

3684

FILE_READ liest asynchron eine Datei aus dem Flash-Speicher. Gleichzeitig können IEC-Tasks ungehindert weiterlaufen.

© ifm electronic gmbh

Parameter der Eingänge

3685

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt Datei wird geschlossen
PATH	DWORD	Adresse des Pfadnamens der Datei, Zuweisung eines Strings mit dem kompletten Pfadnamen über ADR-Operator, z.B.: `'/home/project/data.txt'` Änderung nur erlaubt während ENABLE = FALSE **ACHTUNG!** Falsche Adresse kann die Systemdaten zerstören!
READ	BOOL	Flanke FALSE ⇨ TRUE (nur 1 Zyklus lang): Datei-Lesen starten Startimpuls gilt nur, wenn vorherige Operation beendet wurde. FALSE: im weiteren Programmablauf
INSIZE	DWORD	Anzahl der zu lesenden Bytes
INPOS	DWORD	Startwert, ab wo in der Datei gelesen werden soll
LIMIT_CHAR	BYTE	Zeichenbegrenzung: Wert = 0: INSIZE = Anzahl zu lesender Bytes Wert nicht 0: Alle Bytes bis zum nächsten Begrenzer in der Datei lesen. Wenn kein Begrenzer gefunden, dann INSIZE = Anzahl zu lesender Bytes Der Begrenzer selbst wird nicht gespeichert (Text-Modus)
DATA	DWORD	Adresse, wohin die gelesenen Daten gespeichert werden sollen. Dazu ADR-Operator verwenden! **ACHTUNG!** Falsche Adresse kann die Systemdaten zerstören!

Parameter der Ausgänge

3886

Parameter	Datentyp	Beschreibung
RESULT	BYTE	Status der Operation: 0 = nicht freigegeben oder nicht aktiv 1 = Datenübertragung erfolgreich beendet, nächste Operation kann gestartet werden 2 = Baustein aktiv (Datenübertragung) 3 = Fehler; Baustein muss beendet (disabled) werden
FILESIZE	DWORD	Anzahl der Bytes in der Datei
OUTSIZE	DWORD	Anzahl der gelesenen Bytes: 0 = Datei-Ende
OUTPOS	DWORD	Offset, wo Datei nach dem Lesen beginnt

© ifm electronic gmbh

FILE_WRITE

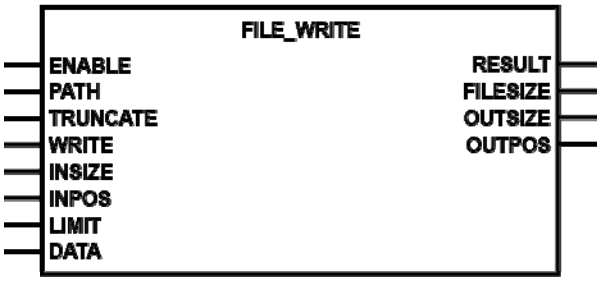
3674

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_SysLibFile_Async_Vxxyyzz.Lib`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Symbol in CoDeSys:



Beschreibung

3687

FILE_WRITE speichert asynchron eine Datei im Flash-Speicher.

© ifm electronic gmbh

Parameter der Eingänge

3688

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein wird ausgeführt Die angegebene Datei wird geöffnet FALSE: Baustein wird nicht ausgeführt Datei wird geschlossen, sobald alle Daten gespeichert sind
PATH	DWORD	Adresse des Pfadnamens der Datei Änderung nur erlaubt während ENABLE = FALSE **ACHTUNG!** Falsche Adresse kann die Systemdaten zerstören!
TRUNCATE	BOOL	TRUE: Datei wird überschrieben, wenn ENABLE = FALSE⇨TRUE⇨FALSE FALSE: Datei bleibt erhalten Neue Daten werden angehängt oder eingefügt Änderungen werden erst übernommen bei ENABLE = FALSE
WRITE	BOOL	Flanke FALSE ⇨ TRUE (nur 1 Zyklus lang): Datei-Schreiben starten Startimpuls gilt nur, wenn vorherige Operation beendet wurde. FALSE: im weiteren Programmablauf
INSIZE	DWORD	Anzahl der zu speichernder Bytes
INPOS	DWORD	Startwert, ab wo in die Datei geschrieben werden soll
LIMIT_CHAR	BYTE	Zeichenbegrenzung: Wert = 0: INSIZE = Anzahl zu speichernder Bytes Wert nicht 0: Alle Bytes bis zum nächsten Begrenzer in der Datei speichern. Wenn kein Begrenzer gefunden, dann INSIZE = Anzahl zu speichernder Bytes Der Begrenzer selbst wird nicht gespeichert (Text-Modus)
DATA	DWORD	Adresse, woher die zu speichernden Daten gelesen werden sollen **ACHTUNG!** Falsche Adresse kann die Systemdaten zerstören!

Parameter der Ausgänge

3689

Parameter	Datentyp	Beschreibung
RESULT	BYTE	Status der Operation: 0 = nicht freigegeben oder nicht aktiv 1 = Datenübertragung erfolgreich beendet, nächste Operation kann gestartet werden 2 = Baustein aktiv (Datenübertragung) 3 = Fehler; Baustein muss beendet (disabled) werden
FILESIZE	DWORD	Anzahl der Bytes in der Datei
OUTSIZE	DWORD	Anzahl der gespeicherten Bytes: 0 = Datei-Ende
OUTPOS	DWORD	Offset, wo Datei nach dem Speichern beginnt

© ifm electronic gmbh

10 SPS-Zyklus optimieren

Inhalt:

Zykluszeit steuern 241

8609

Hier zeigen wir Ihnen Funktionen zum Optimieren des SPS-Zyklus.

10.1 Zykluszeit steuern

Inhalt:

GET_PENALTY 242
OPT_SCHEDULER 244

3142

© ifm electronic gmbh

10.1.1 GET_PENALTY

3592

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_CR10nn_Vxxyyzz.Lib`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Symbol in CoDeSys:

Beschreibung

3611

GET_PENALTY liefert folgende Aussagen:

- Welches tatsächliches Intervall hat der aktuelle Task?
- Ist der aktuelle Task zurzeit mit einer Strafzeit (Penalty) belegt?

Dazu sollte dieser FB in jedem Task aufgerufen werden. Die jeweils gelieferten Aussagen geben dem Applikations-Programmierer wertvolle Hinweise für weitere Optimierungen der Task-Konfiguration.

Empfehlungen für Task-Konfigurationen

3612

► In CoDeSys über dem Reiter [Ressourcen] > [Task-Konfiguration] doppelklicken.

> **Voreingestellt** erscheinen hier folgende Parameter:

Task	ruft auf	Prio	Intervall
PLC_PRG_TASK	PLC_PRG	1	zyklisch alle 10 ms
VISU_TASK	MAINTARGETVISU_PAINT_CODE	15	zyklisch alle 200 ms
VISU_INPUT_TASK	MAINTARGETVISU_INPUT_CODE	14	zyklisch alle 50 ms

Wir empfehlen, Folgendes zu beachten:

- Der optimierte Scheduler ist noch kein echter Realtime-Scheduler.
- Alle IEC-Task-Intervalle im optimierten Scheduler sollen ein Vielfaches von 20 ms haben. Linux arbeitet intern mit einem 10-ms-Raster. Wegen der Penalty-Regelung, die auf die Grenze ½ Intervall prüft, ist das kleinste, sinnvoll einstellbare Intervall 20 ms; praxistauglicher sind Intervalle ≥ 40 ms.
- Speziell beim 3S-Scheduler bei zu langsamem Systemverhalten das Intervall eher erhöhen anstatt senken.
- Beim optimierten Scheduler für Tasks, die oft mit Penalty bestraft werden, ebenfalls das Intervall erhöhen.
- Bei Priorität 1...15 ist der optimierte Scheduler aktiv.
  Bei Priorität > 15 ist der 3S-Scheduler aktiv.
- Visualisierungs-Tasks sollten immer mit Priorität > 15 betrieben werden. Ansonsten wäre ein vernünftiger Betrieb nicht mehr möglich.

© ifm electronic gmbh

Tabelle: Vorschlag zur Änderung der voreingestellten Task-Parameter:

Task	ruft auf	Prio	Intervall
PLC_PRG_TASK	PLC_PRG	1	zyklisch alle 20 / 40 ms
VISU_TASK	MAINTARGETVISU_PAINT_CODE	18	zyklisch alle 200 / 400 ms
VISU_INPUT_TASK	MAINTARGETVISU_INPUT_CODE	17	zyklisch alle 50 ms

Weitere Tasks lassen sich einfach ergänzen und parametrieren (→ CoDeSys-Handbuch).

Parameter der Eingänge

3613

Parameter	Datentyp	Beschreibung
TASK_INTERVAL_PRESET	WORD	Wert aus der Task-Konfiguration übertragen, z.B. Intervall = 40 ms ⇨ Wert = 40

Parameter der Ausgänge

3614

Parameter	Datentyp	Beschreibung
TASK_INTERVAL	WORD	Anzeige aktuelle Intervalldauer in [ms].
PENALTY	BOOL	TRUE: Der Task wird zurzeit mit einer Strafzeit von 2x Intervall bestraft. ► Das Intervall verlängern (in der CoDeSys-Task-Konfiguration UND in diesem Baustein), bis keine Strafzeit mehr vergeben wird. FALSE: Der Task läuft ohne Strafzeit ► Prüfen, ob sich das Intervall senken lässt, ohne dass Penalty eine Strafzeit vergibt.

© ifm electronic gmbh

10.1.2 OPT_SCHEDULER

3589

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_CR10nn_Vxxyyzz.Lib`

Für folgende Geräte verfügbar: (ab Betriebssystem Version 4.41)
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Symbol in CoDeSys:

Beschreibung

3597

OPT_SCHEDULER schaltet den "Optimierten Task-Scheduler" ein oder aus.

► Vor dem Geräte-Neustart ein Bootprojekt erzeugen und speichern!

Die Änderung ist erst nach Geräte-Neustart (Versorgungsspannung Aus-Ein) wirksam.

HINWEIS

Wenn in einem Gerät der optimierte Task-Scheduler ein- oder ausgeschaltet ist, dann bleibt er unverändert so – auch bei einer Projekt-Änderung. Wir empfehlen daher dringend:

► Diesen FB in allen Projekten für PDM und PDMcompact aufrufen und in jedem Projekt den optimierten Task-Scheduler gezielt ein- oder ausschalten.

HINWEIS

Auch der optimierte Scheduler kann Ressourcen-Probleme nicht kompensieren. Bei zu hoher CPU-Last kommt es auch hier zu Laufzeit-Schwankungen (Jitter) und langen Zykluszeiten (z.B. langsame Bildwechsel).

Der optimierte Scheduler sollte nur sorgfältig geplant und konfiguriert erfolgen. Bei ungünstigem Einsatz kann das Laufzeitverhalten der Applikation noch schlechter sein als beim Standard-Scheduler.

© ifm electronic gmbh

Was ist ein Scheduler?

3604

Der Scheduler verteilt die zur Verfügung stehende CPU-Zeit an die Anwendungen (Tasks):
- diverse Linux-Tasks,
- diverse System-Tasks (z.B. Drehgeber, Tasten),
- IEC-Tasks (z.B. PLC_PRG, Visu_Task).

Ein IEC-Task wird definiert über:
- sein Intervall (wie häufig aufgerufen),
- seine Priorität (wie wichtig eingestuft).

Tasks mit relativ niedriger Priorität aber langer Laufzeit können von wichtigeren Tasks unterbrochen werden. Solche Unterbrechungen wirken sich meist sehr negativ auf das Systemlaufverhalten aus.

Was macht der 3S-Task-Scheduler?

3605

Beim in CoDeSys integrierten Scheduler sind Intervall und Priorität nur bedingt unter der Kontrolle des Anwenders.

- IEC-Tasks werden unter Umständen unterbrochen.
- Die Gesamt-Zykluszeit wird oft verlängert.
- Die eingestellten Intervalle werden oft nicht eingehalten (Jitter-Wirkung).
- Der Prozess wird dadurch instabil.

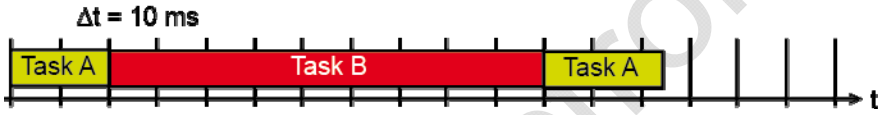


Bild: Unterbrechung eines IEC-Tasks durch einen anderen Task.
Im Beispiel wird der IEC-Task A von Task B unterbrochen. Erst nach Ablauf von Task B kann Task A zu Ende bearbeitet werden. Dadurch wird die Gesamtzeit von Task A länger als ohne Unterbrechung.

© ifm electronic gmbh

Was macht der "Optimierte Task-Scheduler"?

3606

Der optimierte Task-Scheduler unterscheidet sich vom 3S-Task-Scheduler wie folgt:

- IEC-Tasks werden nicht unterbrochen.
- Die vom Task benötigte Gesamtzeit wird meist kürzer.
- Die eingestellten Intervalle werden besser eingehalten (weniger Jitter).
- Der Prozess wird dadurch stabiler.

Bild 1: Anderer Task wird an laufenden IEC-Task angehängt:
Task B startet erst nach Ablauf von Task A.

Nachteil: Ein überlanger Task, der sehr häufig gestartet wird, kann das ganze System ausbremsen oder sogar lahmlegen:

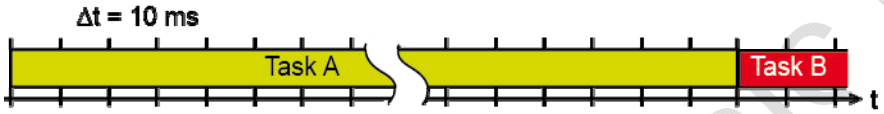


Bild 2: Überlanger Task
Wegen überlanger Laufzeit von Task A käme der Task B erst sehr verzögert an die Reihe.

Deshalb muss der Applikations-Programmierer die Prioritäten und Intervalle an der laufenden Aplikation testen (z.B. mit Hilfe von *GET_PENALTY*, → Seite 242) und optimieren.

© ifm electronic gmbh

Das Penalty-Konzept

3607

Penalty (engl.) = Strafe

Der optimierte Task-Scheduler setzt ein Penalty-Konzept ein. Damit werden Tasks seltener gestartet, wenn sie in der Vergangenheit die ihnen zugemessene Laufzeit überschritten haben.

Das Penalty-Konzept prüft in jedem Zyklus die jeweilige tatsächliche Laufzeit der Tasks. Sobald ein Task länger als 50 % der ihm zugemessenen Laufzeit (= konfiguriertes Intervall) benötigt, wird dieser Task zur Strafe für (mindestens) dem Doppelten der zugemessenen Laufzeit nicht mehr aufgerufen. In dieser Zeit kommen andere Tasks an die Reihe. Sobald der "bestrafte" Task wieder im Rahmen der vorgegebenen Zeit bleibt, entfällt die Strafzeit wieder.

Beispiel:

In der CoDeSys-Task-Konfiguration sind für die Tasks A...C jeweils max. 20-ms-Intervalle eingeplant worden. Task A und B haben eine hohe Priorität, Task C eine niedrige. Anfangs benötigen alle Tasks max. die Hälfte davon, also bis zu 10 ms (→ Bild 1). Es ist somit alles in Ordnung, alle Tasks laufen in dichter Folge:

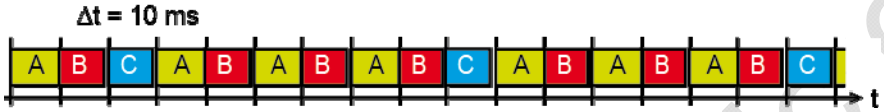


Bild 1: Tasks A...C dauern jeweils max. 10 ms.

► Bei einer Aktion in der Applikation wurde der Arbeitsumfang für Task A umfangreicher (→ Bild 2):

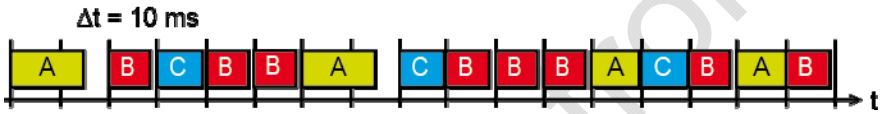


Bild 2: Task A benötigt zwei Zyklen lang jeweils mehr als 10 ms. Das Penalty-Konzept bestraft Task A dafür jedesmal mit einer Strafzeit von 2x 20 ms. In diesen 40 ms können andere Tasks (ggf. auch mehrfach) arbeiten. Sobald sich Task A wieder an die Grenze von ½ Intervall (= 10 ms) hält, entfällt seine Strafzeit wieder.

Für Task A besteht offensichtlich Optimierungsbedarf.

© ifm electronic gmbh

Parameter der Eingänge

3609

Parameter	Datentyp	Beschreibung
INIT	BOOL	TRUE (steigende Flanke): Baustein wird initialisiert (nur 1 Zyklus lang) FALSE: im weiteren Programmablauf
SET	BOOL	TRUE: aktiviert den optimierten Scheduler FALSE: aktiviert den 3S-Standard-Scheduler

Die Änderung ist erst nach Geräte-Neustart (Versorgungsspannung Aus-Ein) wirksam.

Parameter der Ausgänge

3610

Parameter	Datentyp	Beschreibung
OP_SCHEDULER	BOOL	TRUE: Der optimierte Scheduler läuft FALSE: Der 3S-Standard-Scheduler läuft
MODE_CHANGED	BOOL	TRUE (nur 1 Zyklus lang): Der Scheduler wurde umgeschaltet Geräte-Neustart erforderlich!

© ifm electronic gmbh

11 LED, Buzzer, Visualisierung

Inhalt:

LED ansteuern 249
Visualisierung verwalten 251
Eingabefelder in Visualisierung bearbeiten 271

8615

Hier zeigen wir Ihnen folgende Funktionen:

- Ansteuern von LED
- Ansteuern des Buzzer
- Verwalten der Visualisierung

11.1 LED ansteuern

Inhalt:

PDM_LED_OFF 250

8230

Hier zeigen wir Ihnen Funktionen zum Ansteuern der LED in diesem Gerät.

© ifm electronic gmbh

11.1.1 PDM_LED_OFF

3166

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_CRnnnn_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Symbol in CoDeSys:

Beschreibung

3270

Mit PDM_LED_OFF lassen sich alle Tasten-LEDs gleichzeitg ausschalten. Dagegen steuern die LED-Funktionen in PDM_MAIN jede einzelne Tasten-LED.

Parameter der Eingänge

3271

Parameter	Datentyp	Beschreibung
INIT	BOOL	TRUE (steigende Flanke): Baustein wird initialisiert (nur 1 Zyklus lang) FALSE: im weiteren Programmablauf
OFF	BOOL	TRUE: Alle LEDs werden abgeschaltet FALSE: Ursprüngliche LED-Beschaltung ist wieder aktiv

© ifm electronic gmbh

11.2 Visualisierung verwalten

Inhalt:

PDM_MAIN ... 252
PDM_MAIN_MAPPER ... 255
PDM_PAGECONTROL ... 259
CURRENTVISU_TO_NO ... 261
PAGENO_TO_CURRENTVISU ... 263
Bibliothek Instrumente ... 264

8617

Hier zeigen wir Ihnen Funktionen zum Verwalten von Visualisierungen.

© ifm electronic gmbh

11.2.1 PDM_MAIN

3149

Baustein-Typ = Programm (PRG)

Enthalten in Bibliothek: `ifm_CRnnnn_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Symbol in CoDeSys:

Beschreibung

3258

PDM_MAIN enthält folgende wichtige Funktionen für die Initialisierung des PDM360:

- PDM_OPEN_IO
- PDM_KEY
- PDM_LED
- PDM_ENC_DATA

Sie sollten PDM_MAIN in jedes PDM-Projekt integrieren, andernfalls müssen die oben beschriebenen Funktionen einzeln abgearbeitet werden.

HINWEIS

Wird PDM_MAIN eingesetzt, dürfen die oben genannten Einzel-Funktionen nicht eingesetzt werden.

Werden oben genannte Einzel-Funktionen eingesetzt, darf PDM_MAIN nicht eingesetzt werden.

Parameter der Eingänge

3259

Parameter	Datentyp	Beschreibung
INIT	BOOL	TRUE (steigende Flanke): Baustein wird initialisiert (nur 1 Zyklus lang) FALSE: im weiteren Programmablauf

© ifm electronic gmbh

Globale Variable dieses Programms

3260

Alle Variablen dieses Programms sind in den Globalen Variablen der Bibliothek abgelegt.

Name	Datentyp	Beschreibung
F1...F6 *)	BOOL	Status der Funktionstasten (gedrückt = TRUE).
KEY_CHANGE	BOOL	Impuls: Tastenstatus hat sich verändert.
RT_F1...RT_F6 ¹)	BOOL	Impuls: Funktionstaste betätigt.
FT_F1...FT_F6 ¹)	BOOL	Impuls: Funktionstaste losgelassen.
RT_ESC *)	BOOL	Impuls: Taste [ESC] betätigt.
FT_ESC *)	BOOL	Impuls: Taste [ESC] losgelassen.
KEYCODE	BYTE	Tastenstatus bitcodiert: **Bit** 7 6 5 4 3 2 1 0 **Wert** 128 64 32 16 8 4 2 1 **Taste** -- -- F1 F2 F3 F4 F5 F6
LED1...LED6 ¹), LED_OK	BOOL	TRUE = LED eingeschaltet. FALSE = LED ausgeschaltet. Die LED-Helligkeit ist abhängig von LED_MAX_VALUE und LED_NIGHT_VALUE.
LED_MAX_VALUE	BYTE	LED-Helligkeit für Normalbetrieb (0...100 %).
LED_NIGHT_VALUE	BYTE	LED-Helligkeit für Nachtbetrieb (0...100 %).
LED_NIGHT	BOOL	LED-Helligkeit im Nachtbetrieb.
BACKLIGHT ²)	BYTE	Display-Helligkeit (0...100 %; Werkseinstellung = 100 %). Bei zu geringen Werten ist der Bildschirm nicht mehr lesbar!
CONTRAST ³)	BYTE	Display-Kontrast (0...100%, Werkseinstellung = 50 %). Bei zu geringen oder zu hohen Werten ist der Bildschirm nicht mehr lesbar!
RESET_ENCODER	BOOL	Drehgeber-Zählerwert zurücksetzen.
INC	WORD	Aktueller Drehgeber-Zählerwert (0...65.535).
ENC_PUSHED	BOOL	Drehgeber-Taste ist gedrückt.
ENC_CHANGED	BOOL	Drehgeber-Zählerwert hat sich geändert.
ENCODER_UP	BOOL	Drehgeber wurde im Uhrzeigersinn gedreht. Drehgeber-Zählerwert wurde erhöht.
ENCODER_DOWN	BOOL	Drehgeber wurde entgegen Uhrzeigersinn gedreht. Drehgeber-Zählerwert wurde gesenkt.
PDM_FILE_OPEN_ERROR	BOOL	Initialisierungsfehler.
MAPPER_STATE	DWORD	Wert >1 ⇨ PDM befindet sich im Eingabemodus.
MAPPER_TYPE	DWORD	Typ des Eingabeelementes: 10 = inkrementelle Eingabe. 18 = Positionseingabe. 32 = Eingabe über Zahlenblock.

¹) Die Geräte PDM360compact: CR1052, CR1053, CR1055, CR1056 enthalten nur 3 Tasten und 3 LEDs.

²) Sobald der Anwender den Wert von BACKLIGHT ändert, wird dieser sofort vom Gerät übernommen und spannungsausfallsicher gespeichert.

© ifm electronic gmbh

Nur CR1051: Um die Lebensdauer der Leuchte zu erhöhen, wird im Temperaturbereich -10...5 °C die Helligkeit automatisch durch das Betriebsystem reduziert. Erst oberhalb der Geräte-Innentemperatur von +5 °C wird der in BACKLIGHT eingestellte Sollwert übernommen.
Geräte-Innentemperatur lesen → *PDM_TEMPERATURE*

³) gilt nur für PDM360compact: CR1052, CR1053, CR1055, CR1056.

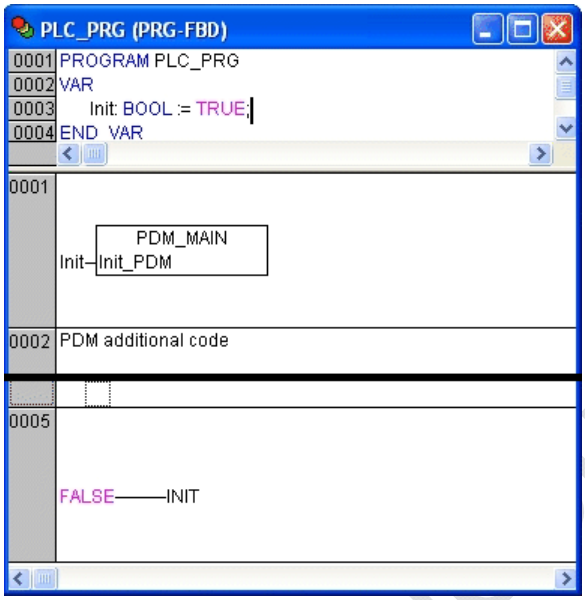
*) Wegen Kompatibilität zu PDM360smart werden von der Taste [F1/ESC] die Signale für [F1] und [ESC] getrennt ausgewertet.

PDM_MAIN einsetzen

3261

Sie sollten PDM_MAIN in eines der ersten Netzwerke des Applikations-Programms einbinden.

Der Eingang INIT darf nur im ersten Programmzyklus auf TRUE gesetzt werden.



Beispiel: Einsatz von PDM_MAIN

Wenn Sie prüfen wollen, ob PDM_MAIN erfolgreich initialisiert ist:

► Variable PDM_FILE_OPEN_ERROR abfragen.

> Wenn PDM_FILE_OPEN_ERROR = TRUE ⇨ fehlerhafte Initialisierung (z.B. Init_PDM wurde nicht zurückgesetzt).

© ifm electronic gmbh

11.2.2 PDM_MAIN_MAPPER

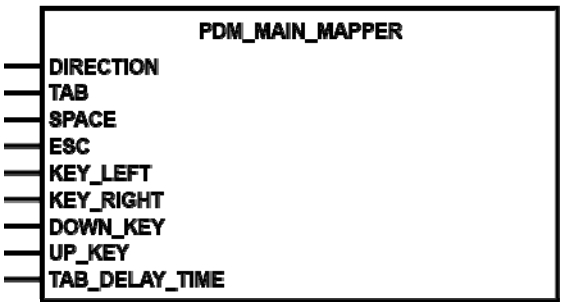
3151

Baustein-Typ = Programm (PRG)

Enthalten in Bibliothek: `ifm_CRnnnn_Vxxyyzz.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Symbol in CoDeSys:



Beschreibung

3262

Das Programm PDM_MAIN_MAPPER ist die Schnittstelle zwischen CoDeSys-Tastaturkommandos für die Bedienung der Visualisierung und dem Laufzeitsystem des PDM. Durch Setzen/Rücksetzen der einzelnen Eingänge werden die Eingaben der PC-Tastatur emuliert.

© ifm electronic gmbh

Parameter der Eingänge

3263

Parameter	Datentyp	Beschreibung
DIRECTION	BOOL	Entspricht der PC-Taste [Umschalt] oder [Shift]. Sinnvoll mit dem Eingang TAB: TRUE: Die Markierung wechselt zum vorhergehenden Element FALSE: Die Markierung wechselt zum nächsten Element
TAB	BOOL	TRUE (Impuls, erstmalig): Markieren des ersten Elements der Elementliste, für das eine Eingabe konfiguriert ist TRUE (Impuls, weiterer) und DIRECTION=FALSE: Weiterschalten zum nächsten eingabefähigen Element TRUE (Impuls, weiterer) und DIRECTION=TRUE: Zurückschalten zum vorhergehenden eingabefähigen Element FALSE: diese Funktion wird nicht ausgeführt
SPACE	BOOL	TRUE (Impuls, erster): Selektiertes Visualisierungselement betätigen. Je nach gewähltem Eingabemodus kann dann im Eingabefeld navigiert werden. TRUE (Impuls, zweiter): Eingabe beenden; (neuen) Wert ins PDM schreiben FALSE: diese Funktion wird nicht ausgeführt
ESC	BOOL	TRUE (Impuls): Editiermodus abbrechen; Wert nicht verändern FALSE: diese Funktion wird nicht ausgeführt
KEY_LEFT	BOOL	TRUE (Impuls) und Eingabemodus=Position: Cursor im Eingabefeld um eine Position nach links verschieben FALSE: diese Funktion wird nicht ausgeführt
KEY_RIGHT	BOOL	TRUE (Impuls) und Eingabemodus=Position: Cursor im Eingabefeld um eine Position nach rechts verschieben FALSE: diese Funktion wird nicht ausgeführt
DOWN_KEY	BOOL	TRUE (Impuls) und Eingabemodus=Schrittweite: Wert im Eingabefeld um die angegebene Schrittweite mindern FALSE: diese Funktion wird nicht ausgeführt
UP_KEY	BOOL	TRUE (Impuls) und Eingabemodus=Schrittweite: Wert im Eingabefeld um die angegebene Schrittweite erhöhen FALSE: diese Funktion wird nicht ausgeführt
TAB_DELAY_TIME	TIME	Zeitverzögerung für den Eingang TAB Typische Werte: 250...400 ms Wert etwas größer einstellen als die Intervall-Zeit VISU_TASK

© ifm electronic gmbh

Beispiele: PDM_MAIN_MAPPER

3264

Damit Eingabeelemente über die Mapperfunktion gesteuert werden können, müssen sie zuvor konfiguriert werden:

► Eingabeelement markieren.

► Mit Rechtsklick das Kontextmenü öffnen.

► Menüpunkt [Konfigurieren] wählen.

► Kategorie [Eingabe] wählen.

► Sinnvolle Werte für [Position] und [Schrittweite] eintragen (→ folgende Beispiele):

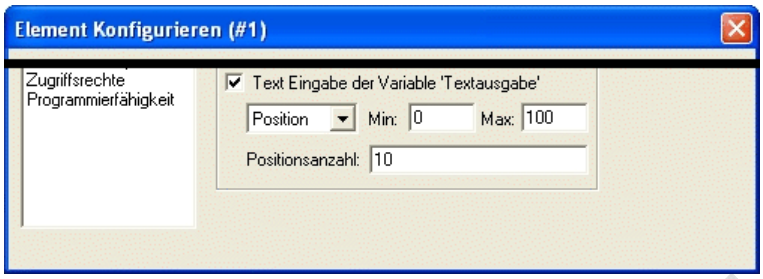


Bild: Beispiel für Eingabemodus Position

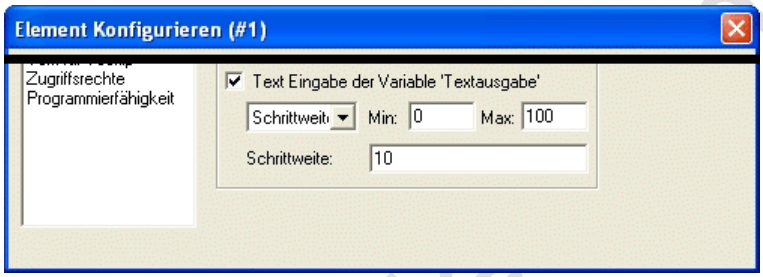
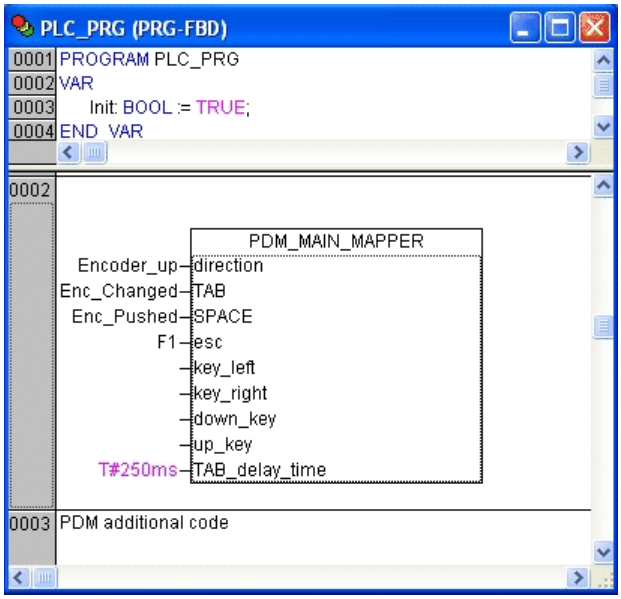


Bild: Beispiel für Eingabemodus Schrittweite

► Sie sollten PDM_MAIN_MAPPER in eines der ersten Programmnetzwerke einbinden.

© ifm electronic gmbh

Im folgenden Beispiel wird der Drehgeber für die Bedienung der aktiven Visualisierungselemente genutzt:



© ifm electronic gmbh

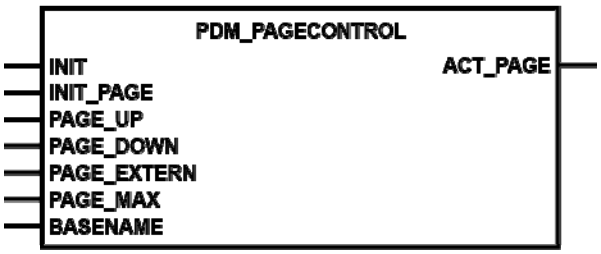
11.2.3 PDM_PAGECONTROL

3186

Baustein-Typ = Programm (PRG)

Enthalten in Bibliothek:	Für folgende Geräte verfügbar:
`ifm_PDM_UTIL_Vxxyyzz.LIB`	- PDM360: CR1050, CR1051 - PDM360compact: CR1052, CR1053, CR1055, CR1056
`ifm_PDMng_UTIL_Vxxyyzz.LIB`	- PDM360NG: CR108n
`ifm_PDMsmart_UTIL_Vxxyyzz.LIB`	- PDM360smart: CR1070, CR1071

Symbol in CoDeSys:



Beschreibung

3294

PDM_PAGECONTROL steuert den Aufruf bestimmter Visualisierungsseiten. Der Aufruf und die Rückgabe der Visualisierungsseiten erfolgt in CoDeSys über die Systemvariable CurrentVisu (vom Typ STRING[40]).

Mit dem Programm kann wahlweise eine bestimmte Visualisierungsseite aufgerufen oder schrittweise in den Visualisierungen geblättert werden.

Das Programm lässt sich optimal nutzen, wenn die Namen aller Visualisierungen dem gleichen Schema entsprechen, also einer Kombination aus einem Basisnamen, gefolgt von einer 5-stelligen Zahl (ab Bibliotheks-Version V04.00.07; davor: 3-stellig *)).

Beispiel BASENAME = `PAGE`:
Visualisierungsname = `PAGE00001`, `PAGE00002`, `PAGE00003`, usw.

Für den Basisnamen sind 1...35 Großbuchstaben (keine Sonderzeichen) zulässig. Die Nummerierung der Visualisierungen sollte lückenlos erfolgen. Das Programm setzt den endgültigen Visualisierungsnamen aus dem Parameter BASENAME und der Nummer zusammen oder liest die Nummer aus dem aktuellen Visualisierungsnamen aus und stellt sie im Ausgangsparameter ACT_PAGE zur Verfügung.

Anstatt die Visualisierungen mit Basisnamen und laufender Nummer zu benennen, kann jede Visualisierung auch individuell benannt werden, z.B.: `SERVICE1`, `MOTORDATA2`, `CONFIGURATION3`. Die Programmierung ist in diesem Fall aber aufwendiger, weil Basisname und Visualisierungsnummer einzeln zugewiesen werden müssen. Ein schrittweises Blättern ist nur noch sehr eingeschränkt möglich.

Verwenden Sie als BASENAME den Buchstaben `P`, dann ist Ihr Programm kompatibel mit den **ifm**-Templates.

***) Beachten Sie die neue 5-stellige Nummerierung auch bei der Namensgebung Ihrer bereits bestehenden Visualisierungsseiten!**

© ifm electronic gmbh

Parameter der Eingänge

3293

Parameter	Datentyp	Beschreibung
INIT	BOOL	TRUE (nur 1 Zyklus lang): Display wird initialisiert mit der in INIT_PAGE angegebenen Initialisierung FALSE: im weiteren Programmablauf
INIT_PAGE	WORD	Visualisierungsnummer, die mit INIT aufgerufen werden soll
PAGE_UP	BOOL	Flanke FALSE ⇨ TRUE: inkrementiert die Visualisierungsnummer
PAGE_DOWN	BOOL	Flanke FALSE ⇨ TRUE: dekrementiert die Visualisierungsnummer
PAGE_EXTERN	WORD	Angegebene Visualisierungsseite wird direkt aufgerufen (unabhängig von PAGE_UP / PAGE_DOWN) Sobald PAGE_EXTERN = ACT_PAGE, dann PAGE_EXTERN wieder auf "0" setzen!
PAGE_MAX	WORD	Maximale Anzahl der anwählbaren Visualisierungsseiten
BASENAME	STRING[35]	Gemeinsamer Namensbestandteil der Visualisierungsseite. Die Nummerierung der Visualisierungsseiten erfolgt durch die Namensgebung, z.B. "P00001". Hierbei gelten: - "P" = BASENAME (nur Großbuchstaben!) - "00001" = Visualisierungsnummer (5-stellig!)

Parameter der Ausgänge

3295

Parameter	Datentyp	Beschreibung
ACT_PAGE	WORD	aktuelle Visualisierungsnummer

© ifm electronic gmbh

11.2.4 CURRENTVISU_TO_NO

3188

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: bis `ifm_PDM_UTIL_V040006.Lib`

Ab V04.00.07 ist diese Funktion durch *PDM_PAGECONTROL* (→ Seite 259) ersetzt.

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Symbol in CoDeSys:

Beschreibung

3296

CURRENTVISU_TO_NO ist eine Hilfsfunktion zur Ermittlung der Visualisierungsnummer. Dieser FB wird für alte Projekte benötigt, die mit der Bibliothek PDM_LIB_B.LIB erstellt wurden.

Der FB konvertiert den String der Systemvariablen CurrentVisu in die entsprechende Elementnummer des Arrays. Am Ausgang steht immer die Nummer der aktuellen Visualisierungsseite an. Der String enthält die Liste aller verwendeten Visualisierungsnamen. Die Reihenfolge in dieser Liste bestimmt die resultierende Visualisierungsnummer.

Beispiel:

PAGE	VISUNAMES
001	STARTPAGE001
002	PARAMETERS001
003	PARAMETERS002
004	OPERATIONMODE001
005	ACTUALVALUES001
006	ACTUALVALUES002
007	ERRORS001

Der zugehörige String lautet: 'STARTPAGE001','PARAMETERS001','PARAMETERS002', 'OPERATIONMODE001','ACTAULVALUES001','ACTAULVALUES002','ERRORS001'

HINWEIS

In CURRENTVISU_TO_NO ist VISUNAMES vom TYPE ARRAY[1..200] of STRING[40].

In der PDM_LIB_B ist VISUNAMES vom TYPE ARRAY[1..255] of STRING.

In neuen Projekten verwenden Sie besser *PDM_PAGECONTROL* (→ Seite 259).

© ifm electronic gmbh

Parameter der Eingänge

3297

Parameter	Datentyp	Beschreibung
VISUNAMES	ARRAY[1...200] OF STRING(40)	Es wird ein STRING ARRAY mit 255 Elementen von 1...200 erwartet. In diesem Array müssen die exakten Namen der Visualisierungsseiten eingetragen sein.

Beispiel:

ARRAY[1...200] OF STRING[40]:='MAIN','SUB1','SUB2','ALARM1','MAINTENANCE','CONTROL','PAGE1','PAGE2', ...

Parameter der Ausgänge

3298

Parameter	Datentyp	Beschreibung
PAGE	BYTE	aktuelle Visualisierungsnummer

© ifm electronic gmbh

11.2.5 PAGENO_TO_CURRENTVISU

3191

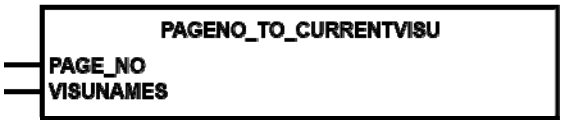
Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: bis `ifm_PDM_UTIL_V040006.Lib`

Ab V04.00.07 ist dieser FB durch *PDM_PAGECONTROL* (→ Seite 259) ersetzt.

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056

Symbol in CoDeSys:



Beschreibung

3299

PAGENO_TO_CURRENTVISU ist eine Hilfsfunktion zum Aufruf einer Visualisierungsseite. Diese FB wird für alte Projekte benötigt, die mit der Bibliothek PDM_LIB_B.LIB erstellt wurden.

Der FB konvertiert die übergebene Seitennummer in den entsprechenden Namen der Visualisierungsseite, übergibt diesen String an die Systemvariable CurrentVisu und ruft die Visualisierungsseite auf.

HINWEIS

In PAGENO_TO_CURRENTVISU ist VISUNAMES vom TYPE ARRAY[1..200] of STRING[40].

In der PDM_LIB_B ist VISUNAMES vom TYPE ARRAY[1..255] of STRING.

In neuen Projekten verwenden Sie besser *PDM_PAGECONTROL* (→ Seite 259).

Parameter der Eingänge

3300

Parameter	Datentyp	Beschreibung
PAGE_NO	BYTE	Nummer der gewünschten Visualisierungsseite
VISUNAMES	ARRAY[1...200] OF STRING[40]	Es wird ein STRING ARRAY mit 255 Elementen von 1...200 erwartet. In diesem Array müssen die exakten Namen der Visualisierungsseiten eingetragen sein.

Beispiel:

ARRAY[1..200] OF STRING[40]:='MAIN','SUB1','SUB2','ALARM1','MAINTENANCE','CONTROL','PAGE1','PAGE2', ...

© ifm electronic gmbh

11.2.6 Bibliothek Instrumente

Inhalt:

CONTROL_ANALOGCLOCK 266
SCALE_LED_GRAF 267
SCALE_METER 269

3354

Einbinden von fertigen Visualisierungs-Elementen

Diese Bibliothek steht weiterhin zur Verfügung, um mit älteren Applikationen kompatibel zu bleiben. Wir empfehlen stattdessen – wegen deutlich besserer Darstellung – den Einsatz von Hintergrund-Bitmaps.
Siehe auch: Darstellbare → *CoDeSys-Visualisierungs-Elemente*

Die Bibliothek `Instrumente_x.LIB` bietet eine Anzahl von vorgefertigten Visualisierungs-Elementen. Diese können Sie direkt in Ihre Visualisierungsseiten über [Einfügen] > [Visualisierung] einbinden. Die Visualisierungs-Elemente sind so aufgebaut, dass die aktiven Elemente über Platzhalter animiert werden können. Dazu werden die Platzhalter direkt mit einer Variablen aus dem Applikations-Programm verknüpft. Weiter Informationen finden Sie in der CoDeSys-Onlinehilfe unter "Platzhalter in der Visualisierung".

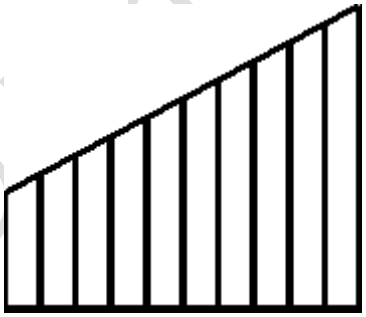
Die Bibliothek enthält folgende Funktionen:

- *CONTROL_ANALOGCLOCK*, → Seite 266
  zeigt die aktuelle Uhrzeit auf dem Zifferblatt einer Analoguhr:

Analog_Clock

- *SCALE_LED_GRAF*, → Seite 267
  zeigt Eingangswerte als eine 10-stellige, werteabhängige LED-Zeile:

Visu = Bargraf_LED10_H

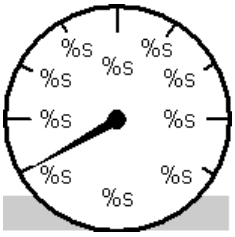
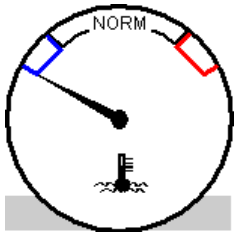


Visu = Bargraf_LED10_H2

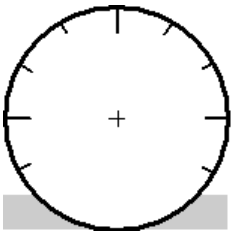
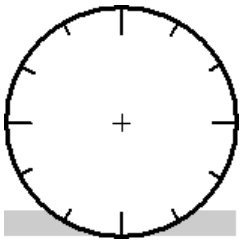
Visu = Bargraf_LED10_V

© electronic gmbh

- *SCALE_METER*, → Seite 269
  zeigt Eingangswerte als kreisförmige Messgeräte-Skala:

			
METER_NO = 1	METER_NO = 2	METER_NO = 3	METER_NO = 4
Visu = Meter1	Visu = Meter2	Visu = Meter3	Visu = Meter4

- Zusätzlich bietet die Bibliothek als Visualisierung 2 neutrale Skalen:

	
Visu = ClockFace1	Visu = ClockFace2

© ifm electronic gmbh

CONTROL_ANALOGCLOCK

3366

Baustein-Typ = Programm (PRG)

Enthalten in Bibliothek: `Instrumente_x.LIB`

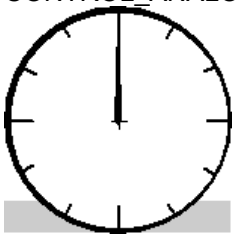
Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360smart: CR1070, CR1071
- PDM360NG: CR108n

Symbol in CoDeSys:

Beschreibung

3378

CONTROL_ANALOGCLOCK zeigt die aktuelle Uhrzeit auf dem Zifferblatt einer Analoguhr:



Parameter der Eingänge

3379

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv
PDM_RTC	DT	Systemzeit und Datum aus SysRtcGetTime

© ifm electronic gmbh

SCALE_LED_GRAF

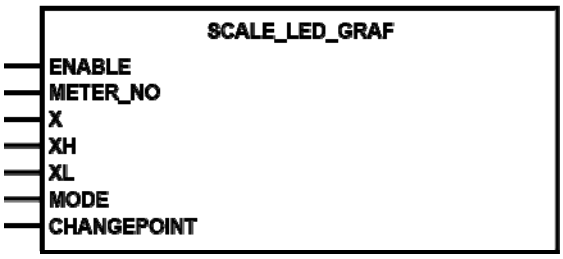
3369

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `Instrumente_x.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360smart: CR1070, CR1071
- PDM360NG: CR108n

Symbol in CoDeSys:

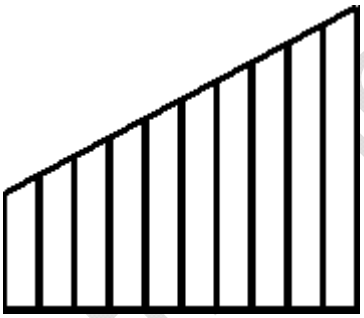


Beschreibung

3381

SCALE_LED_GRAF zeigt Eingangswerte als eine 10-stellige, werteabhängige LED-Zeile, z.B. eine der 3 Visualisierungen aus dieser Bibliothek:

Visu = Bargraf_LED10_H



Visu = Bargraf_LED10_H2

Visu = Bargraf_LED10_V

Der FB bildet einen Eingangswert relativ zu einem definierten Wertebereich ab.

© ifm electronic gmbh

Parameter der Eingänge

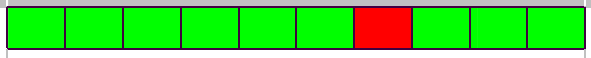
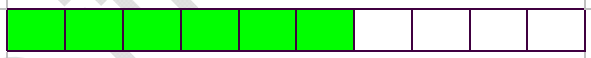
3382

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv
X	INT	Eingangswert
XH	INT	Obere Grenze des Wertebereichs
XL	INT	Untere Grenze des Wertebereichs
MODE	BYTE	Betriebsart der LED-Kette Wertebereich = 0...10
CHANGEPOINT	BYTE	Farbwechsel-Punkt bei MODE = 9 oder 10 Wertebereich = 0...10

Betriebsart der LED-Kette

3383

Alle Variablen dieses Programms sind in den Globalen Variablen der Bibliothek abgelegt.

Mode	LED-Kette	Beschreibung
1		rotes Einzel-Segment auf grünem Leuchtband
2		grünes Einzel-Segment auf rotem Leuchtband
3		rotes Einzel-Segment
4		grünes Einzel-Segment
5		rote Segmentkette auf grünem Leuchtband
6		grüne Segmentkette auf rotem Leuchtband
7		rote Segmentkette
8		grüne Segmentkette
9		rote Segmentkette mit Farbwechsel-Punkt (hier CHANGEPOINT = 5)
10		grüne Segmentkette mit Farbwechsel-Punkt (hier CHANGEPOINT = 7)

© ifm electronic gmbh

SCALE_METER

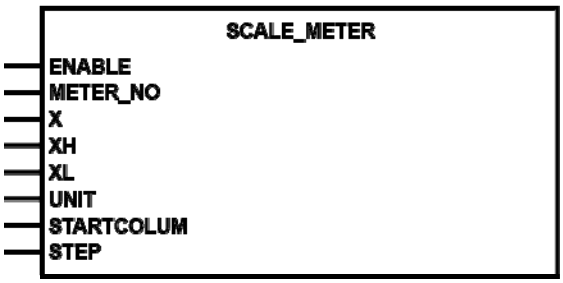
3372

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `Instrumente_x.LIB`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360smart: CR1070, CR1071
- PDM360NG: CR108n

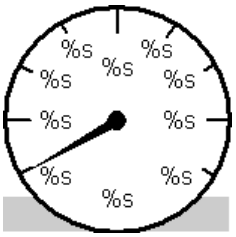
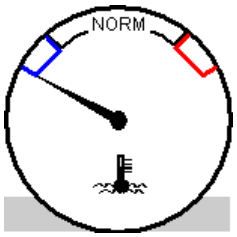
Symbol in CoDeSys:



Beschreibung

3384

SCALE_METER zeigt Eingangswerte als kreisförmige Messgeräte-Skala:

			
METER_NO = 1	METER_NO = 2	METER_NO = 3	METER_NO = 4
Visu = Meter1	Visu = Meter2	Visu = Meter3	Visu = Meter4

Der FB bildet einen Eingangswert relativ zu einem definierten Wertebereich ab.

In der Visualisierung Meter1 dient "%s" als Platzhalter für die parametrierten Werte und Einheit. In den anderen Visualisierungen gibt es keine oder keine definierbaren Skalenwerte.

© ifm electronic gmbh

Parameter der Eingänge

3385

Parameter	Datentyp	Beschreibung
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv
METER_NO	BYTE	1 = Meter1 = 270°-Skala mit definierbaren Werten und Einheit 2 = Meter2 = 270°-Skala 3 = Meter3 = Tankanzeige 4 = Meter4 = Temperaturanzeige
X	INT	Eingangswert
XH	INT	Obere Grenze des Wertebereichs
XL	INT	Untere Grenze des Wertebereichs
UNIT	STRING[6]	Für METER_NO = 1: Maßeinheit in der Skala (Text)
STARTCOLUM	INT	Startwert der Skala, z.B.: 10 = Die Skala beginnt bei 10
STEP	INT	Schrittweite der Skala, z.B.: 10 = Skalawerte für 10, 20, 30, ...

© ifm electronic gmbh

11.3 Eingabefelder in Visualisierung bearbeiten

Inhalt:

INPUT_INT 272
INPUT_DINT 277
INPUT_REAL 280
INIT_VAR_RETAIN_INT 283
INIT_VAR_RETAIN_DINT 285
INIT_VAR_RETAIN_REAL 287

10088

Hier zeigen wir Ihnen Funktionen zum Bearbeiten von Eingabefeldern in Visualisierungen.

© ifm electronic gmbh

11.3.1 INPUT_INT

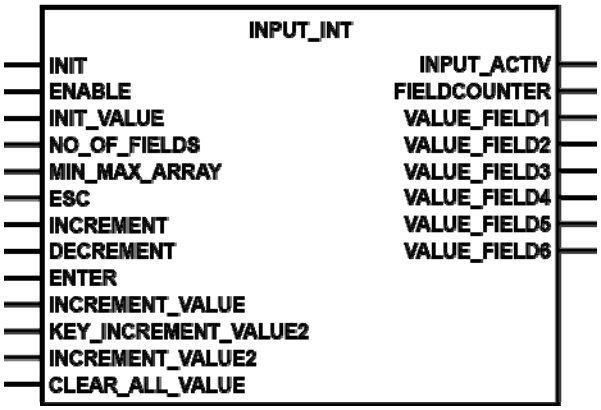
3220

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_INPUT_Vxxyyzz.Lib`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:



Beschreibung

3328

INPUT_INT unterstützt bis zu 6 Eingabefelder, die in der Visualisierung 6 verschiedenen Objekten zugewiesen werden können. Die Eingabeobjekte werden mit den internen Variablen FIELD1...FIELD6, COLOR_FIELD1...COLOR_FIELD6 und VALUE_FIELD1...VALUE_FIELD6 parametriert.

Der FB ist eine Alternative zur in CoDeSys integrierten Eingabefunktion und gegenüber der CoDeSys-Eingabe flexibler.

Der FB enthält keine Visualisierungselemente.

© ifm electronic gmbh

Parameter der Eingänge

3329

Parameter	Datentyp	Beschreibung
INIT	BOOL	TRUE (nur 1 Zyklus lang): Setzt den Fokus auf das erste Eingabefeld FALSE: im weiteren Programmablauf
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv
INIT_VALUE	ARRAY[1...6] OF INT	Initialisierungswerte für die Eingabefelder beim erstmaligen Aufruf
NO_OF_FIELDS	BYTE	Anzahl der benötigten Eingabefelder: Wertebereich = 1...6
MIN_MAX_ARRAY	ARRAY[1...12] OF INT	Legt die Eingabegrenzen fest. Index 1 = min. Wert Feld 1 Index 2 = max. Wert Feld 1 Index 3 = min. Wert Feld 2 Index 4 = max. Wert Feld 2 usw.
ESC	BOOL	TRUE: Setzt den Eingabewert zurück auf den zuletzt gültigen Wert FALSE: diese Funktion wird nicht ausgeführt
INCREMENT	BOOL	Flanke FALSE ⇨ TRUE: Inkrementiert den Eingabewert um den Wert in INCREMENT_VALUE (oder INCREMENT_VALUE2)
DECREMENT	BOOL	Flanke FALSE ⇨ TRUE: Dekrementiert den Eingabewert um den Wert in INCREMENT_VALUE (oder INCREMENT_VALUE2)
ENTER	BOOL	1. Flanke FALSE ⇨ TRUE: > aktiviert Eingabemodus für das Eingabefeld > wechselt Feldfarbe (nur Color-Gerät) > Cursor blinkt 2. Flanke FALSE ⇨ TRUE: > Eingabemodus für das Eingabefeld zurücksetzen > editierte Werte in die Ausgangsvariablen VALUE_FIELD1...VALUE_FIELD6 schreiben
INCREMENT_VALUE	INT	Wenn KEY_INCREMENT_VALUE2 = FALSE: um diesen Betrag wird der Wert im aktiven Eingabefeld bei INCREMENT / DECREMENT verändert
KEY_INCREMENT_VALUE2	BOOL	TRUE: der Wert im aktiven Eingabefeld ändert sich bei INCREMENT / DECREMENT um den Betrag in INCREMENT_VALUE2 FALSE: der Wert im aktiven Eingabefeld ändert sich bei INCREMENT / DECREMENT um den Betrag in INCREMENT_VALUE
INCREMENT_VALUE2	INT	Wenn KEY_INCREMENT_VALUE2 = TRUE: um diesen Betrag wird der Wert im aktiven Eingabefeld bei INCREMENT / DECREMENT verändert
CLEAR_ALL_VALUE	BOOL	TRUE: alle internen und externen Eingabewerte auf "0" setzen FALSE: diese Funktion wird nicht ausgeführt

© ifm electronic gmbh

Parameter der Ausgänge

3330

Parameter	Datentyp	Beschreibung
INPUT_ACTIV	BOOL	aktuelle Visualisierungsnummer
FIELDCOUNTER	BYTE	Nummer des gewählten Eingabefeldes
VALUE_FIELD1	INT	externer Eingabewert, wird erst mit der 2. steigenden Flanke von ENTER übernommen
...	INT	
VALUE_FIELD6	INT	

Interne Parameter

3331

Parameter	Datentyp	Beschreibung
FIELD1	INT	Setzt den Fokus für das gewählte Eingabefeld. Als Fokus kann ein beliebiges Visualisierungsobjekt verwendet werden. Die Variable FIELDx muss in der Visualisierung diesem Objekt zugewiesen werden. Ist die Eingabe aktiv, wird FIELDx getaktet.
...	INT	
FIELD6	INT	
COLOR_FIELD1	INT	Setzt die Alarmfarbe des Eingabeobjektes (nur PDM color). Die Variable muss in der Visualisierung zugewiesen werden.
...	INT	
COLOR_FIELD6	INT	
VALUE_FIELD_1	INT	Interner Eingabewert, der während der Eingabe sofort sichtbar ist und mit der zweiten steigenden Flanke von ENTER an VALUE_FIELDx übergeben wird. Die Variable muss in der Visualisierung zugewiesen werden.
...	INT	
VALUE_FIELD_6	INT	

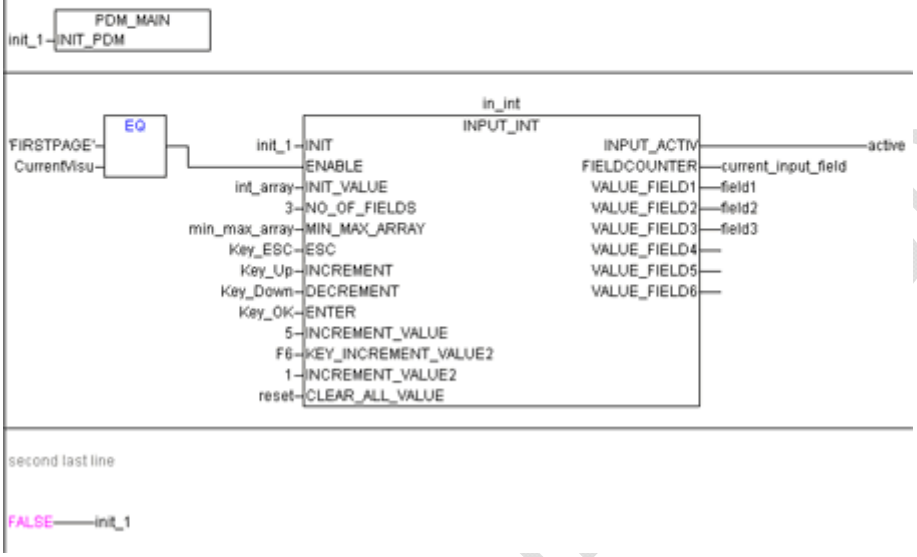
© ifm electronic gmbh

Beispiel: INPUT_INT mit 3 Eingabeelementen

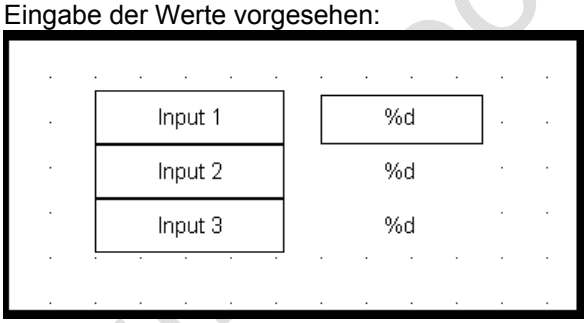
3332

► In das Applikations-Programm den gewünschten Funktionsblock einbinden und die Ein- und Ausgänge mit den Parametern und Variablen verknüpfen:

```
PROGRAM PLC_PRG
VAR
    init_1: BOOL := TRUE ;
    cycletime:WORD;
    max_cycletime:WORD;
    reset_max:BOOL;
    in_int: INPUT_INT;
    int_array: ARRAY [1..6] OF INT := 1,2,3;
    min_max_array: ARRAY [1..12] OF INT:=1,10,2,20,3,30;
```

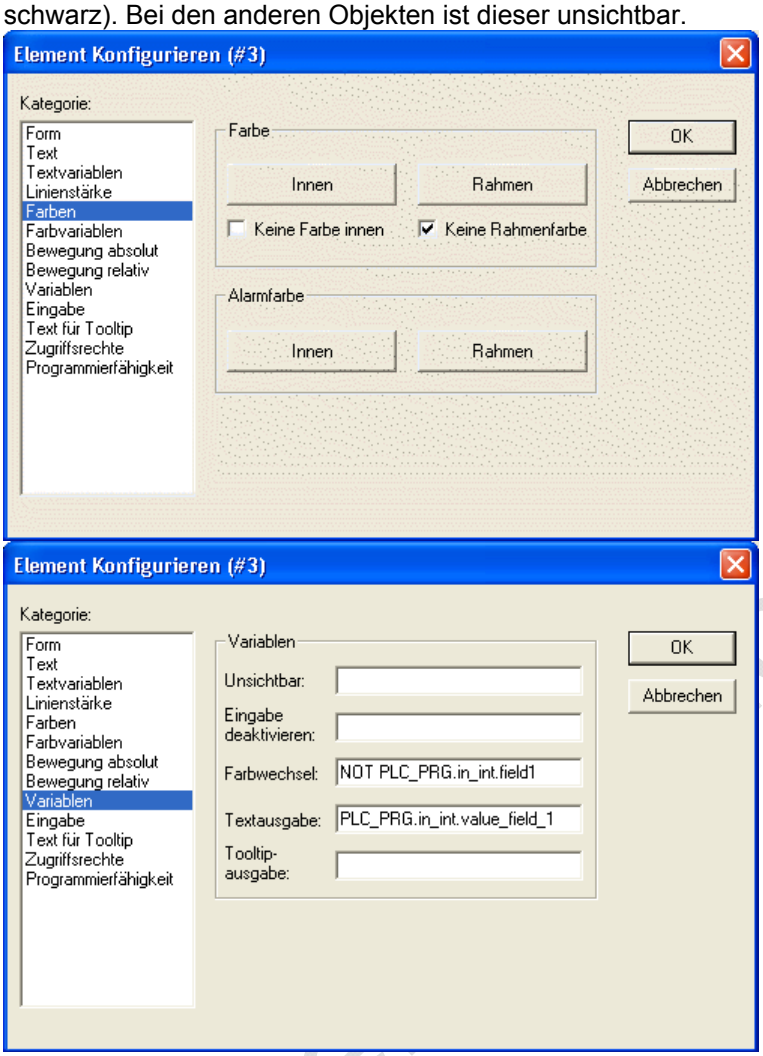


► Die gewünsche Visualisierungsseite erzeugen. Im Beispiel sind die drei rechten Elemente für die Eingabe der Werte vorgesehen:



© ifm electronic gmbh

► Durch Konfiguration der Eingabeelemente mit den internen Parametern FIELD1...FIELD3 wird ein Rahmen bei dem ausgewählen Eingabeobjekt angezeigt (FIELDn = TRUE, Alarmfarbe Rahmen = schwarz). Bei den anderen Objekten ist dieser unsichtbar.



► Als Textausgabe-Variable die interne Variable VALUE_FIELD_n wählen.

> Bei laufender Applikation wird das ausgewählte Eingabefeld durch den Rahmen angezeigt.

> Nach erster Flanke FALSE [TRUE am FB-Eingang ENTER:
der Objekt-Rahmen blinkt und der Eingabewert kann editiert werden.

> Nach zweiter Flanke FALSE [TRUE am FB-Eingang ENTER:
der Wert wird übernommen und der Rahmen wird wieder statisch angezeigt.

© ifm electronic gmbh

11.3.2 INPUT_DINT

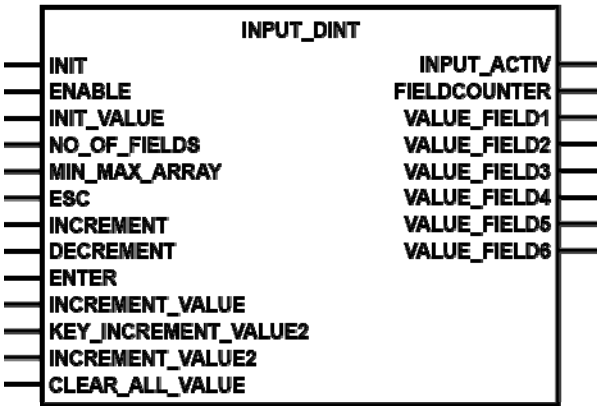
3223

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_INPUT_Vxxyyzz.Lib`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:



Beschreibung

3333

INPUT_DINT unterstützt bis zu 6 Eingabefelder, die in der Visualisierung 6 verschiedenen Objekten zugewiesen werden können. Die Eingabeobjekte werden mit den internen Variablen FIELD1...FIELD6, COLOR_FIELD1...COLOR_FIELD6 und VALUE_FIELD1...VALUE_FIELD6 parametriert.

Der FB ist eine Alternative zur in CoDeSys integrierten Eingabefunktion und gegenüber der CoDeSys-Eingabe flexibler.

Der FB enthält keine Visualisierungselemente.

© ifm electronic gmbh

Parameter der Eingänge

3334

Parameter	Datentyp	Beschreibung
INIT	BOOL	TRUE (nur 1 Zyklus lang): Setzt den Fokus auf das erste Eingabefeld FALSE: im weiteren Programmablauf
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv
INIT_VALUE	ARRAY[1..6] OF DINT	Initialisierungswerte für die Eingabefelder beim erstmaligen Aufruf
NO_OF_FIELDS	BYTE	Anzahl der benötigten Eingabefelder: Wertebereich = 1...6
MIN_MAX_ARRAY	ARRAY[1..12] OF DINT	Legt die Eingabegrenzen fest. Index 1 = min. Wert Feld 1 Index 2 = max. Wert Feld 1 Index 3 = min. Wert Feld 2 Index 4 = max. Wert Feld 2 usw.
ESC	BOOL	TRUE: Setzt den Eingabewert zurück auf den zuletzt gültigen Wert FALSE: diese Funktion wird nicht ausgeführt
INCREMENT	BOOL	Flanke FALSE ⇨ TRUE: Inkrementiert den Eingabewert um den Wert in INCREMENT_VALUE (oder INCREMENT_VALUE2)
DECREMENT	BOOL	Flanke FALSE ⇨ TRUE: Dekrementiert den Eingabewert um den Wert in INCREMENT_VALUE (oder INCREMENT_VALUE2)
ENTER	BOOL	1. Flanke FALSE ⇨ TRUE: > aktiviert Eingabemodus für das Eingabefeld > wechselt Feldfarbe (nur Color-Gerät) > Cursor blinkt 2. Flanke FALSE ⇨ TRUE: > Eingabemodus für das Eingabefeld zurücksetzen > editierte Werte in die Ausgangsvariablen VALUE_FIELD1...VALUE_FIELD6 schreiben
INCREMENT_VALUE	DINT	Wenn KEY_INCREMENT_VALUE2 = FALSE: um diesen Betrag wird der Wert im aktiven Eingabefeld bei INCREMENT / DECREMENT verändert
KEY_INCREMENT_VALUE2	BOOL	TRUE: der Wert im aktiven Eingabefeld ändert sich bei INCREMENT / DECREMENT um den Betrag in INCREMENT_VALUE2 FALSE: der Wert im aktiven Eingabefeld ändert sich bei INCREMENT / DECREMENT um den Betrag in INCREMENT_VALUE
INCREMENT_VALUE2	DINT	Wenn KEY_INCREMENT_VALUE2 = TRUE: um diesen Betrag wird der Wert im aktiven Eingabefeld bei INCREMENT / DECREMENT verändert
CLEAR_ALL_VALUE	BOOL	TRUE: alle internen und externen Eingabewerte auf "0" setzen FALSE: diese Funktion wird nicht ausgeführt

© ifm electronic gmbh

Parameter der Ausgänge

3335

Parameter	Datentyp	Beschreibung
INPUT_ACTIV	BOOL	aktuelle Visualisierungsnummer
FIELDCOUNTER	BYTE	Nummer des gewählten Eingabefeldes
VALUE_FIELD1	DINT	externer Eingabewert, wird erst mit der 2. steigenden Flanke von ENTER übernommen
...	DINT	
VALUE_FIELD6	DINT	

Interne Parameter

3336

Parameter	Datentyp	Beschreibung
FIELD1	DINT	Setzt den Fokus für das gewählte Eingabefeld. Als Fokus kann ein beliebiges Visualisierungsobjekt verwendet werden. Die Variable FIELDx muss in der Visualisierung diesem Objekt zugewiesen werden. Ist die Eingabe aktiv, wird FIELDx getaktet.
...	DINT	
FIELD6	DINT	
COLOR_FIELD1	DINT	Setzt die Alarmfarbe des Eingabeobjektes (nur PDM color). Die Variable muss in der Visualisierung zugewiesen werden.
...	DINT	
COLOR_FIELD6	DINT	
VALUE_FIELD_1	DINT	Interner Eingabewert, der während der Eingabe sofort sichtbar ist und mit der zweiten steigenden Flanke von ENTER an VALUE_FIELDx übergeben wird. Die Variable muss in der Visualisierung zugewiesen werden.
...	DINT	
VALUE_FIELD_6	DINT	

© ifm electronic gmbh

11.3.3 INPUT_REAL

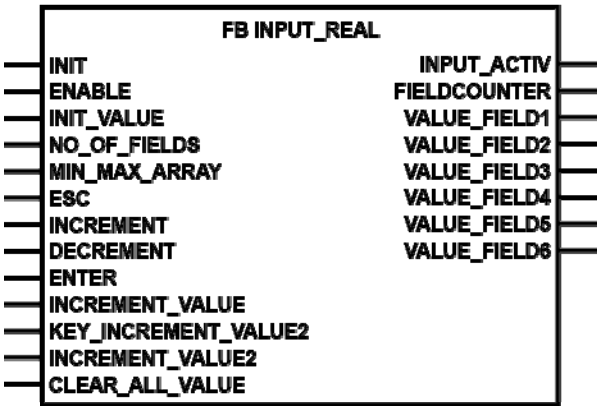
3225

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_INPUT_Vxxyyzz.Lib`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:



Beschreibung

3337

INPUT_REAL unterstützt bis zu 6 Eingabefelder, die in der Visualisierung 6 verschiedenen Objekten zugewiesen werden können. Die Eingabeobjekte werden mit den internen Variablen FIELD1...FIELD6, COLOR_FIELD1...COLOR_FIELD6 und VALUE_FIELD1...VALUE_FIELD6 parametriert.

Der FB ist eine Alternative zur in CoDeSys integrierten Eingabefunktion und gegenüber der CoDeSys-Eingabe flexibler.

Der FB enthält keine Visualisierungselemente.

© ifm electronic gmbh

Parameter der Eingänge

3338

Parameter	Datentyp	Beschreibung
INIT	BOOL	TRUE (nur 1 Zyklus lang): Setzt den Fokus auf das erste Eingabefeld FALSE: im weiteren Programmablauf
ENABLE	BOOL	TRUE: Baustein wird ausgeführt FALSE: Baustein wird nicht ausgeführt > Baustein-Ein- und Ausgänge sind nicht aktiv
INIT_VALUE	ARRAY[1...6] OF REAL	Initialisierungswerte für die Eingabefelder beim erstmaligen Aufruf
NO_OF_FIELDS	BYTE	Anzahl der benötigten Eingabefelder: Wertebereich = 1...6
MIN_MAX_ARRAY	ARRAY[1...12] OF REAL	Legt die Eingabegrenzen fest. Index 1 = min. Wert Feld 1 Index 2 = max. Wert Feld 1 Index 3 = min. Wert Feld 2 Index 4 = max. Wert Feld 2 usw.
ESC	BOOL	TRUE: Setzt den Eingabewert zurück auf den zuletzt gültigen Wert FALSE: diese Funktion wird nicht ausgeführt
INCREMENT	BOOL	Flanke FALSE ⇨ TRUE: Inkrementiert den Eingabewert um den Wert in INCREMENT_VALUE (oder INCREMENT_VALUE2)
DECREMENT	BOOL	Flanke FALSE ⇨ TRUE: Dekrementiert den Eingabewert um den Wert in INCREMENT_VALUE (oder INCREMENT_VALUE2)
ENTER	BOOL	1. Flanke FALSE ⇨ TRUE: > aktiviert Eingabemodus für das Eingabefeld > wechselt Feldfarbe (nur Color-Gerät) > Cursor blinkt 2. Flanke FALSE ⇨ TRUE: > Eingabemodus für das Eingabefeld zurücksetzen. > editierte Werte in die Ausgangsvariablen VALUE_FIELD1...VALUE_FIELD6 schreiben
INCREMENT_VALUE	REAL	Wenn KEY_INCREMENT_VALUE2 = FALSE: um diesen Betrag wird der Wert im aktiven Eingabefeld bei INCREMENT / DECREMENT verändert
KEY_INCREMENT_VALUE2	BOOL	TRUE: der Wert im aktiven Eingabefeld ändert sich bei INCREMENT / DECREMENT um den Betrag in INCREMENT_VALUE2 FALSE: der Wert im aktiven Eingabefeld ändert sich bei INCREMENT / DECREMENT um den Betrag in INCREMENT_VALUE
INCREMENT_VALUE2	REAL	Wenn KEY_INCREMENT_VALUE2 = TRUE: um diesen Betrag wird der Wert im aktiven Eingabefeld bei INCREMENT / DECREMENT verändert
CLEAR_ALL_VALUE	BOOL	TRUE: alle internen und externen Eingabewerte auf "0" setzen FALSE: diese Funktion wird nicht ausgeführt

© ifm electronic gmbh

Parameter der Ausgänge

3339

Parameter	Datentyp	Beschreibung
INPUT_ACTIV	BOOL	aktuelle Visualisierungsnummer
FIELDCOUNTER	BYTE	Nummer des gewählten Eingabefeldes
VALUE_FIELD1	REAL	externer Eingabewert, wird erst mit der 2. steigenden Flanke von ENTER übernommen
...	REAL	
VALUE_FIELD6	REAL	

Interne Parameter

3340

Parameter	Datentyp	Beschreibung
FIELD1	REAL	Setzt den Fokus für das gewählte Eingabefeld. Als Fokus kann ein beliebiges Visualisierungsobjekt verwendet werden. Die Variable FIELDx muss in der Visualisierung diesem Objekt zugewiesen werden. Ist die Eingabe aktiv, wird FIELDx getaktet.
...	REAL	
FIELD6	REAL	
COLOR_FIELD1	REAL	Setzt die Alarmfarbe des Eingabeobjektes (nur PDM color). Die Variable muss in der Visualisierung zugewiesen werden.
...	REAL	
COLOR_FIELD6	REAL	
VALUE_FIELD_1	REAL	Interner Eingabewert, der während der Eingabe sofort sichtbar ist und mit der zweiten steigenden Flanke von ENTER an VALUE_FIELDx übergeben wird. Die Variable muss in der Visualisierung zugewiesen werden.
...	REAL	
VALUE_FIELD_6	REAL	

© ifm electronic gmbh

11.3.4 INIT_VAR_RETAIN_INT

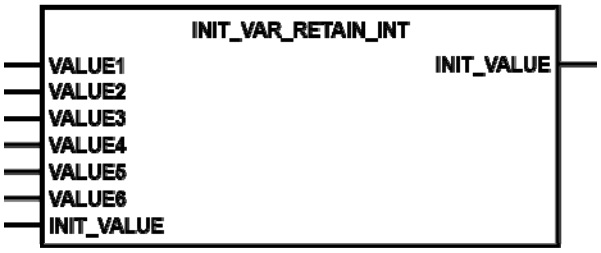
3229

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_INPUT_Vxxyyzz.Lib`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:



Beschreibung

3341

INIT_VAR_RETAIN_INT initialisiert das am Eingang INIT_VALUE angegebene Array des FB *INPUT_INT* (→ Seite 272) mit remanenten Eingabewerten.

Sollen Eingabewerte remanent gespeichert werden, müssen die Werte nach dem Einschalten des Displays initialisiert werden, da sie sonst im ersten Zyklus wieder überschrieben werden. Der FB initialisiert das Array INIT_VALUE des FB INPUT_INT mit den remanenten Eingabewerten.

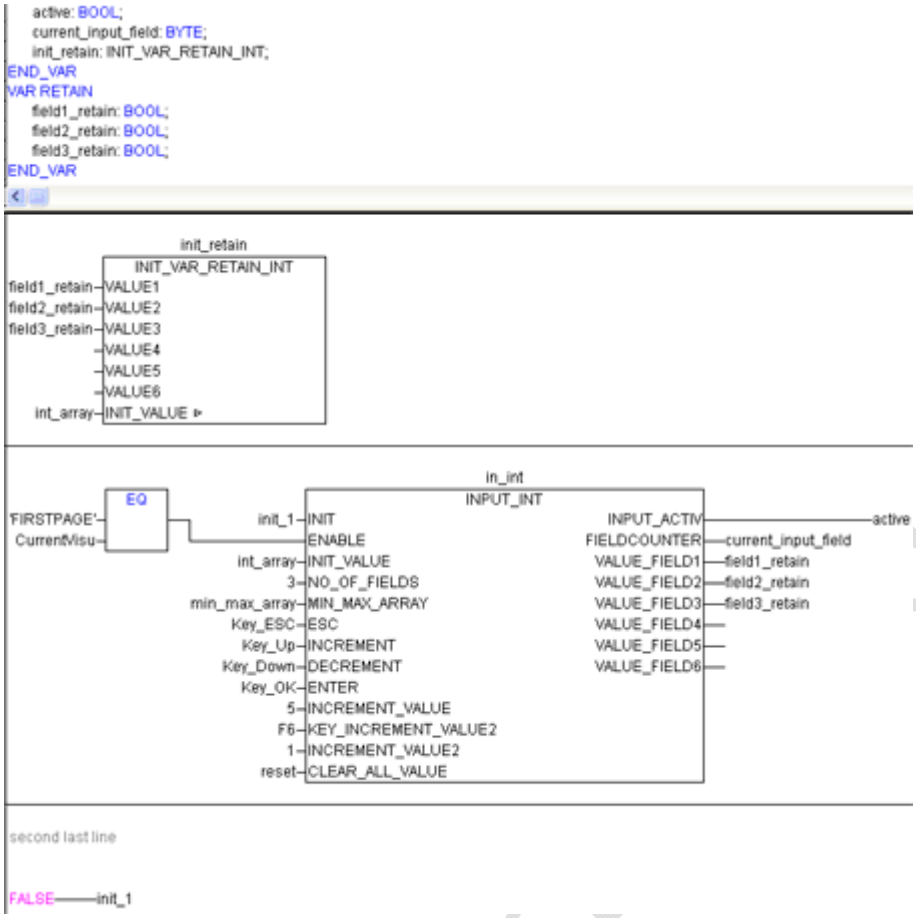
Der Baustein vergleicht die Werte von VALUE1...VALUE6 mit den Werten von INIT_VALUE:

> Sind die Inhalte von VALUE1...VALUE6 = 0, werden VALUE1...VALUE6 in das Array INIT_VALUE geschrieben.

> Sind die Inhalte von VALUE1...VALUE6 nicht 0, werden die Elemente des Arryas INIT_VALUE nach VALUE1...VALUE6 geschrieben.

© ifm electronic gmbh

> Der folgende FB INPUT wird dann im ersten Zyklus mit INIT_VALUE initialisiert.



Beispiel mit INIT_VAR_RETAIN_INT und INPUT_INT

Parameter der Eingänge

3342

Parameter	Datentyp	Beschreibung
VALUE1	INT	An diesem Eingang müssen die als Retain deklarierten Variablen VALUE_FIELDx des FB *INPUT_INT* (→ Seite 272) angegeben werden.
...	INT	
VALUE6	INT	
INIT_VALUE	ARRAY[1...6] OF INT	Sind die Inhalte von VALUE1...VALUE6 nicht 0, werden die Elemente des Arryas INIT_VALUE nach VALUE1...VALUE6 geschrieben.

Parameter der Ausgänge

3343

Parameter	Datentyp	Beschreibung
INIT_VALUE	ARRAY[1...6] OF INT	Sind die Inhalte von VALUE1...VALUE6 = 0, werden VALUE1...VALUE6 in das Array INIT_VALUE geschrieben.

© ifm electronic gmbh

11.3.5 INIT_VAR_RETAIN_DINT

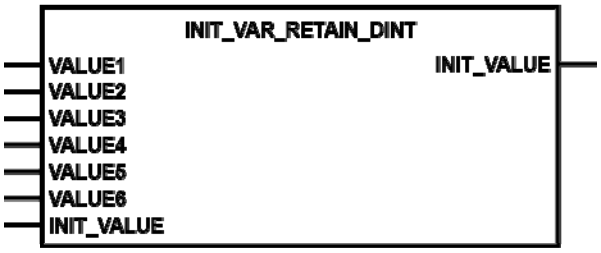
3232

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_INPUT_Vxxyyzz.Lib`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:



Beschreibung

3344

INIT_VAR_RETAIN_DINT initialisiert das am Eingang INIT_VALUE angegebene Array des FB *INPUT_DINT* (→ Seite 277) mit remanenten Eingabewerten.

Sollen Eingabewerte remanent gespeichert werden, müssen die Werte nach dem Einschalten des Displays initialisiert werden, da sie sonst im ersten Zyklus wieder überschrieben werden. Der FB initialisiert das Array INIT_VALUE der Funktion INPUT_DINT mit den remanenten Eingabewerten.

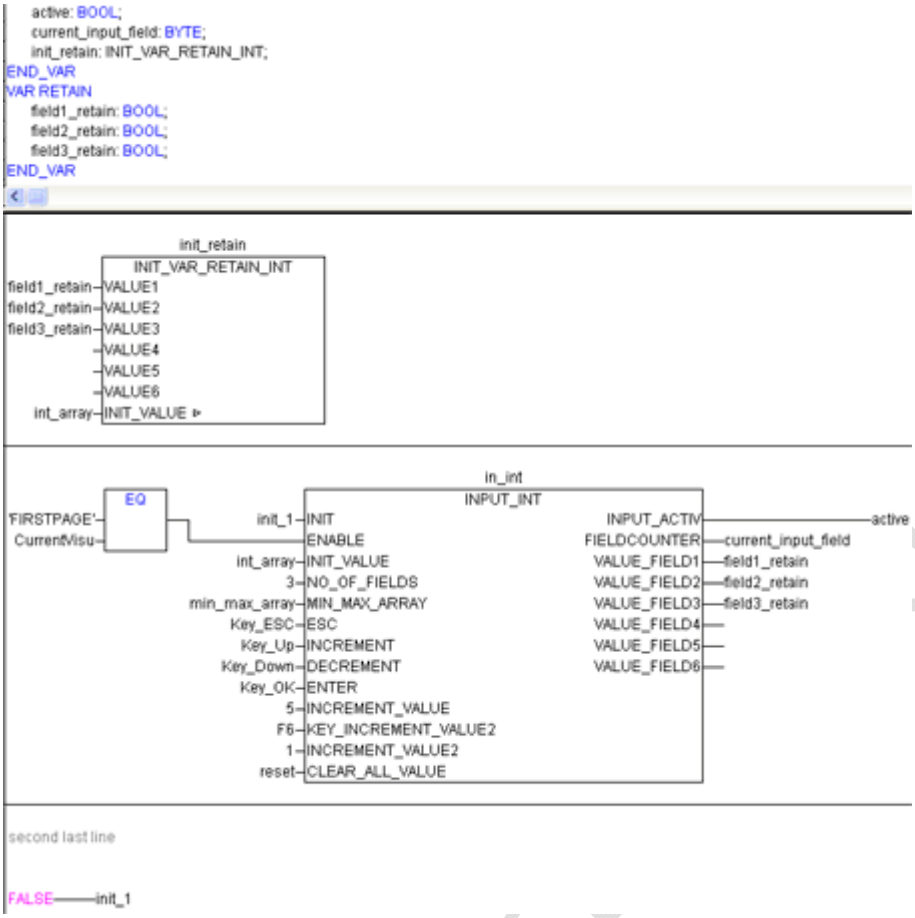
Der Baustein vergleicht die Werte von VALUE1...VALUE6 mit den Werten von INIT_VALUE:

> Sind die Inhalte von VALUE1...VALUE6 = 0, werden VALUE1...VALUE6 in das Array INIT_VALUE geschrieben.

> Sind die Inhalte von VALUE1...VALUE6 nicht 0, werden die Elemente des Arryas INIT_VALUE nach VALUE1...VALUE6 geschrieben.

© ifm electronic gmbh

> Der folgende FB INPUT wird dann im ersten Zyklus mit INIT_VALUE initialisiert.



Beispiel mit INIT_VAR_RETAIN_INT und INPUT_INT

Parameter der Eingänge

3345

Parameter	Datentyp	Beschreibung
VALUE1	DINT	An diesem Eingang müssen die als Retain deklarierten Variablen VALUE_FIELDx dem FB *INPUT_DINT* (→ Seite 277) angegeben werden.
...	DINT	
VALUE6	DINT	
INIT_VALUE	ARRAY[1...6] OF DINT	Sind die Inhalte von VALUE1...VALUE6 nicht 0, werden die Elemente des Arryas INIT_VALUE nach VALUE1...VALUE6 geschrieben.

Parameter der Ausgänge

3346

Parameter	Datentyp	Beschreibung
INIT_VALUE	ARRAY[1...6] OF DINT	Sind die Inhalte von VALUE1...VALUE6 = 0, werden VALUE1...VALUE6 in das Array INIT_VALUE geschrieben.

© ifm electronic gmbh

11.3.6 INIT_VAR_RETAIN_REAL

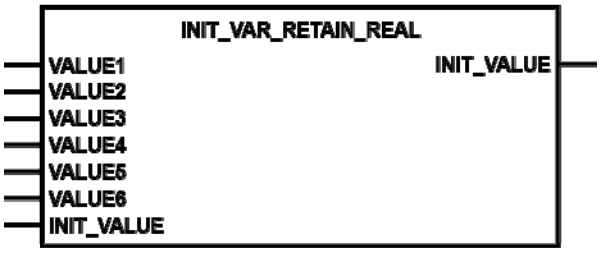
3235

Baustein-Typ = Funktionsblock (FB)

Enthalten in Bibliothek: `ifm_PDM_INPUT_Vxxyyzz.Lib`

Für folgende Geräte verfügbar:
- PDM360: CR1050, CR1051
- PDM360compact: CR1052, CR1053, CR1055, CR1056
- PDM360NG: CR108n

Symbol in CoDeSys:



Beschreibung

3347

INIT_VAR_RETAIN_REAL initialisiert das am Eingang INIT_VALUE angegebene Array des FB *INPUT_REAL* (→ Seite 280) mit remanenten Eingabewerten.

Sollen Eingabewerte remanent gespeichert werden, müssen die Werte nach dem Einschalten des Displays initialisiert werden, da sie sonst im ersten Zyklus wieder überschrieben werden. Der FB initialisiert das Array INIT_VALUE des FB INPUT_REAL mit den remanenten Eingabewerten.

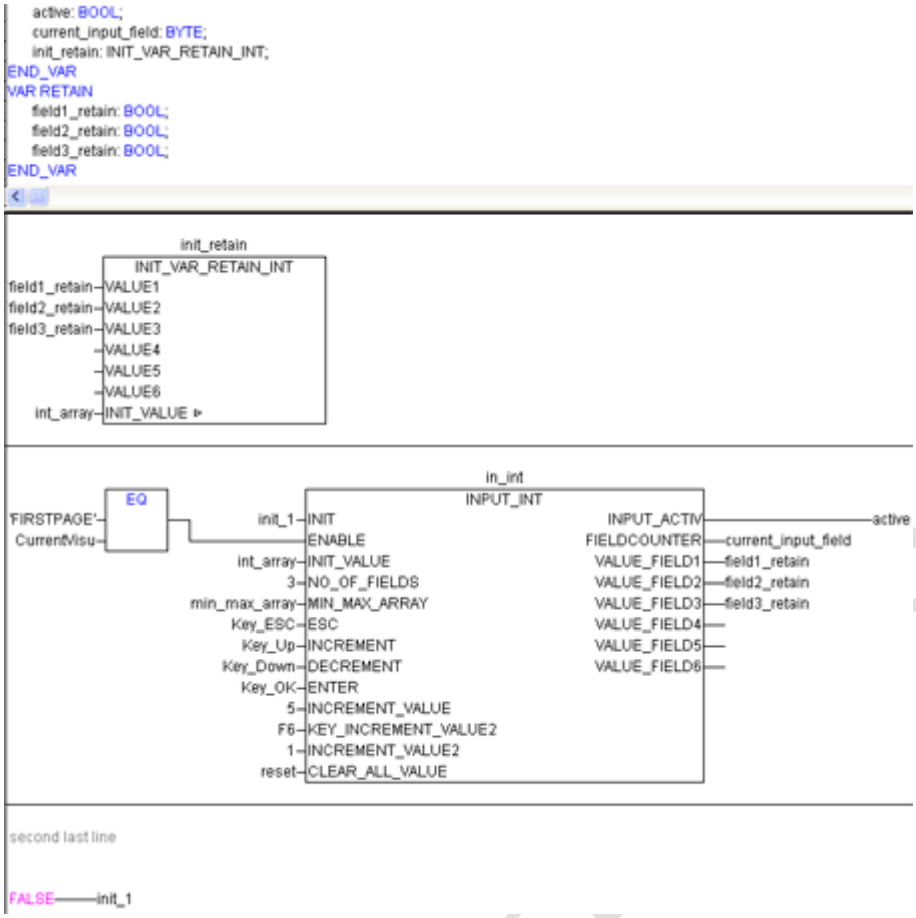
Der Baustein vergleicht die Werte von VALUE1...VALUE6 mit den Werten von INIT_VALUE:

> Sind die Inhalte von VALUE1...VALUE6 = 0, werden VALUE1...VALUE6 in das Array INIT_VALUE geschrieben.

> Sind die Inhalte von VALUE1...VALUE6 nicht 0, werden die Elemente des Arryas INIT_VALUE nach VALUE1...VALUE6 geschrieben.

© ifm electronic gmbh

> Der folgende FB INPUT wird dann im ersten Zyklus mit INIT_VALUE initialisiert.



Beispiel mit INIT_VAR_RETAIN_INT und INPUT_INT

Parameter der Eingänge

3348

Parameter	Datentyp	Beschreibung
VALUE1	REAL	An diesem Eingang müssen die als Retain deklarierten Variablen VALUE_FIELDx dem FB *INPUT_REAL* (→ Seite 280) angegeben werden.
...	REAL	
VALUE6	REAL	
INIT_VALUE	ARRAY[1...6] OF REAL	Sind die Inhalte von VALUE1...VALUE6 nicht 0, werden die Elemente des Arryas INIT_VALUE nach VALUE1...VALUE6 geschrieben.

Parameter der Ausgänge

3349

Parameter	Datentyp	Beschreibung
INIT_VALUE	ARRAY[1...6] OF REAL	Sind die Inhalte von VALUE1...VALUE6 = 0, werden VALUE1...VALUE6 in das Array INIT_VALUE geschrieben.

© ifm electronic gmbh

12 Anhang

Inhalt:

Fehler und Störungen beheben 289
CANopen-Tabellen 290
Visualisierungen im Gerät 304
Übersicht der verwendeten Dateien und Bibliotheken 322

1664

Hier stellen wir Ihnen – ergänzend zu den Angaben in den Datenblättern – zusammenfassende Tabellen zur Verfügung.

12.1 Fehler und Störungen beheben

3109

Hier zeigen wir Ihnen, wie Sie auf bestimmte Fehler und Störungen reagieren können, um das Gerät wieder nutzen zu können.

Wirkung	Ursache	Abhilfe
Nach Neustart vom Setup-Menü aus (= Soft-Reset) bleibt der Hochlauf mit leerem Bild stecken	Seltener Softwarefehler	Hard-Reset über Spannungsversorgung AUS / EIN
Sehr lange Bildwechsel-Zeiten	a) zu viele grafische Elemente im Bild b) zu viele verschiedene Zeichensätze (Fonts) c) zu viele systembelastende Bausteine d) Bausteine zu oft aufgerufen e) zu viele REAL-Variable im Bild	Empfohlene Begrenzungen einhalten! → *Begrenzungen und Programmierhinweise*, → Seite 69
System-Absturz (keine Reaktion)	a) falsche Platzhalter für Variable im CoDeSys-Programm	a) Platzhalter prüfen: z.B. %s (Falsch: %S)
Bildschirm bleibt dunkel	Globale Variable BACKLIGHT zu niedrig eingestellt	a) SETUP-Programm starten und wieder beenden → BACKLIGHT = 90 b) Im Applikations-Programm der BACKLIGHT-Variablen einen höheren Wert zuweisen

© ifm electronic gmbh

12.2 CANopen-Tabellen

Inhalt:

IDs (Adressen) in CANopen 290
Aufbau von CANopen-Meldungen 291
Bootup-Nachricht 296
Netzwerk-Management (NMT) 297
CANopen Error-Code 301

9941

Die folgenden Tabellen informieren Sie über wichtige Werte und Einstellungen der CANopen-Schnittstellen.

12.2.1 IDs (Adressen) in CANopen

3952

In CANopen werden diverse Arten von 'Adressen' (hier: IDs) unterschieden:

- COB-ID
  Der **C**ommunication-**Ob**ject-**Id**entifier adressiert die Nachricht (= das Kommunikationsobjekt) im Geräteverzeichnis. Ein Kommunikationsobjekt besteht aus einem oder mehreren CAN-Nachrichten mit bestimmten Aufgaben, z.B.:
  - PDO (**P**rocess **D**ata **O**bject = Nachrichten-Objekt mit Prozessdaten),
  - SDO (**S**ervice **D**ata **O**bject = Nachrichten-Objekt mit Servicedaten),
  - Emergency (Nachrichten-Objekt mit Notfalldaten),
  - Time (Nachrichten-Objekt mit Zeitangaben) oder
  - Error Control (Nachrichten-Objekt mit Fehlermeldungen).
- CAN-ID
  Der **CAN-Id**entifier definiert netzwerkweit CAN-Nachrichten. Der CAN-ID ist Hauptbestandteil des Arbitration-Feldes eines CAN-Datenübertragungsblocks. Je niedriger der CAN-ID, desto höher die Priorität der Meldung.
- Download-ID
  Der Download-ID bezeichnet den Node-ID für Service-Kommunikation per SDO für den Programm-Download und das Debuggen.
- Node-ID
  Der **Node-Id**entifier ist ein eindeutiger Bezeichner für CANopen-Geräte (Devices) im CAN-Netzwerk. Der Node-ID ist auch Bestandteil einiger vordefinierter Verbindungssätze (→ *Funktions-Code / Predefined Connectionset*, → Seite 293).

Vergleich Download-ID vs. COB-ID:

Controller Programm-Download		CANopen	
Download-ID	**COB-ID SDO**	**Node-ID**	**COB-ID SDO**
1…127	TX: 580_{16} + Download-ID	1…127	TX: 580_{16} + Node-ID
	RX: 600_{16} + Download-ID		RX: 600_{16} + Node-ID

TX = Slave sendet an Master
RX = Slave empfängt von Master

© ifm electronic gmbh

12.2.2 Aufbau von CANopen-Meldungen

Inhalt:

Aufbau des COB-ID 292
Funktions-Code / Predefined Connectionset 293
SDO-Kommando-Bytes 294
SDO-Abbruch-Code 295

9971

Eine CANopen-Meldung besteht aus dem COB-ID und bis zu 8 Bytes Daten:

COB-ID			DLC	Byte 1		Byte 2		Byte 3		Byte 4		Byte 5		Byte 6		Byte 7		Byte 8	
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Details erfahren Sie in den folgenden Kapiteln.

Beachten Sie die umgekehrte Byte-Reihenfolge!

Beispiele:

Wert [hex]	Datentyp	Byte 1		Byte 2		Byte 3		Byte 4		Byte 5		Byte 6		Byte 7		Byte 8	
12	BYTE	1	2	–	–	–	–	–	–	–	–	–	–	–	–	–	–
1234	WORD	3	4	1	2	–	–	–	–	–	–	–	–	–	–	–	–
12345678	DWORD	7	8	5	6	3	4	1	2	–	–	–	–	–	–	–	–

© ifm electronic gmbh

Aufbau des COB-ID

9972

Der erste Teil einer Meldung ist der COB-ID. Aufbau des 11-Bit COB-ID:

Nibble 0				Nibble 1				Nibble 2			
11	10	9	8	7	6	5	4	3	2	1	0
--	**3**	**2**	**1**	**0**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
--	**Funktions-Code**				**Node-ID**						

Der COB-ID besteht aus *Funktions-Code / Predefined Connectionset* (→ Seite 293) und Node-ID.

Beispiel:
Das Kommunikations-Objekt = TPDO1 (TX)
Die Knoten-Nummer des Geräts = 20_{16} = 32_{10}

Berechnung:
Der Funktions-Code für das Kommunikations-Objekt TPDO1 = 3_{16}
Die Wertigkeit des Funktions-Code im 11-Bit-COB-ID = 3_{16} x 80_{16} = 180_{16}
Dazu die Knoten-Nummer (20_{16}) addieren ⇨ der COB-ID lautet: $1A0_{16}$

1				A				0			
3	2	1	0	3	2	1	0	3	2	1	0
0	**0**	**0**	**1**	**1**	**0**	**1**	**0**	**0**	**0**	**0**	**0**
--	3_{16} = 3_{10}				20_{16} = 32_{10}						

© ifm electronic gmbh

Funktions-Code / Predefined Connectionset

9966

Im "CANopen Predefined Connectionset" sind einige Funktions-Codes vorbelegt.

Wenn Sie das Predefined Connectionset verwenden, können Sie ein CANopen-Netzwerk von bis zu 127 Teilnehmern in Betrieb nehmen, ohne dass es zu einer doppelten Vergabe von COB-IDs käme.

Broadcast- oder Multicast-Nachrichten:

Kommunikations-Objekt	Funktions-Code [hex]	COB-ID [hex]	zugehörige Parameter-Objekte [hex]
NMT	0	000	
SYNC	1	080	1005, 1006, 1007, 1028
TIME	2	100	1012, 1013

Punkt-zu-Punkt-Nachrichten:

Kommunikations-Objekt	Funktions-Code [hex]	COB-ID [hex]	zugehörige Parameter-Objekte [hex]
EMERGENCY	1	080 + Node-ID	1014, 1015
TPDO1 (TX)	3	180 + Node-ID	1800
RPDO1 (RX)	4	200 + Node-ID	1400
TPDO2 (TX)	5	280 + Node-ID	1801
RPDO2 (RX)	6	300 + Node-ID	1401
TPDO3 (TX)	7	380 + Node-ID	1802
RPDO3 (RX)	8	400 + Node-ID	1402
TPDO4 (TX)	9	480 + Node-ID	1803
RPDO4 (RX)	A	500 + Node-ID	1403
Default SSDO (TX)	B	580 + Node-ID	1200
Default CSDO (RX)	C	600 + Node-ID	1280
NMT Error Control	E	700 + Node-ID	1016, 1017

TX = Slave sendet an Master
RX = Slave empfängt von Master

SSDO = Server-SDO
CSDO = Client-SDO

© ifm electronic gmbh

SDO-Kommando-Bytes

9968

Aufbau einer SDO-Nachricht:

COB-ID	DLC	Kommando	Index		Sub-Index	Daten *)			
XXX	8	Byte	Byte 0	Byte 1	Byte	Byte 0	Byte 1	Byte 2	Byte 3

*) abhängig von den zu transportierenden Daten

Beachten Sie die umgekehrte Byte-Reihenfolge!

Ein SDO-COB-ID setzt sich wie folgt zusammen:

CANopen	
Node-ID	**COB-ID SDO**
1…127	TX: 580_{16} + Node-ID
	RX: 600_{16} + Node-ID

TX = Slave sendet an Master
RX = Slave empfängt von Master

DLC (Data length code) bezeichnet die Anzahl der Daten-Bytes (bei SDO: DLC = 8).

SDO-Kommando-Bytes:

Kommando hex	dez	Nachricht	Datenlänge	Beschreibung
21	33	Anforderung	mehr als 4 Bytes	Daten an Slave senden
22	34	Anforderung	1…4 Bytes	Daten an Slave senden
23	35	Anforderung	4 Bytes	Daten an Slave senden
27	39	Anforderung	3 Bytes	Daten an Slave senden
2B	43	Anforderung	2 Bytes	Daten an Slave senden
2F	47	Anforderung	1 Byte	Daten an Slave senden
40	64	Anforderung	---	Daten von Slave anfordern
42	66	Antwort	1…4 Bytes	Daten von Slave an Master senden
43	67	Antwort	4 Bytes	Daten von Slave an Master senden
47	71	Antwort	3 Bytes	Daten von Slave an Master senden
4B	75	Antwort	2 Bytes	Daten von Slave an Master senden
4F	79	Antwort	1 Byte	Daten von Slave an Master senden
60	96	Antwort	---	Datentransfer in Ordnung: Empfangsbestätigung von Slave an Master senden
80	128	Antwort	4 Bytes	Datentransfer fehlgeschlagen: Abbruch-Nachricht von Slave an Master senden → Kapitel *SDO-Abbruch-Code*, → Seite 295

© ifm electronic gmbh

SDO-Abbruch-Code

9970

Der SDO-Abbruch-Code gehört NICHT zum Emergency-Telegramm!

Abbruch-Code [hex]	Beschreibung
0503 0000	toggle bit not alternated
0504 0000	SDO protocol timed out
0504 0001	client/server command specifier not valid or unknown
0504 0002	invalid block size (block mode only)
0504 0003	invalid sequence number (block mode only)
0504 0004	CRC error (block mode only)
0504 0005	out of memory
0601 0000	unsupported access to an object
0601 0001	attempt to read a write only object
0601 0002	attempt to write a read only object
0602 0000	object does not exist in the object dictionary
0604 0041	object cannot be mapped to the PDO
0604 0042	the number and length of the objects to be mapped would exceed PDO length
0604 0043	general parameter incompatibility reason
0604 0047	general internal incompatibility in the device
0606 0000	access failed due to an hardware error
0607 0010	data type does not match, length of service parameter does not match
0607 0012	data type does not match, length of service parameter too high
0607 0013	data type does not match, length of service parameter too low
0609 0011	sub-index does not exist
0609 0030	value range of parameter exceeded (only for write access)
0609 0031	value of parameter written too high
0609 0032	value of parameter written too low
0609 0036	maximum value is less than minimum value
0800 0000	general error
0800 0020	data cannot be transferred or stored to the application
0800 0021	data cannot be transferred or stored to the application because of local control
0800 0022	data cannot be transferred or stored to the application because of the present device state
0800 0023	object dictionary dynamic generation fails or no object dictionary is present (e.g. object dictionary is generated from file and generation fails because of an file error)

© ifm electronic gmbh

12.2.3 Bootup-Nachricht

9961

Der CAN-Teilnehmer sendet nach dem Booten einmalig die Bootup-Nachricht:

	Byte 1	Byte 0
hex	700_{16} + Node-ID	NMT-Status
dez	$1\,792_{10}$ + Node-ID	NMT-Status

Somit ist der Teilnehmer im CAN-Netzwerk lauffähig.

Beispiel:
Der Node-ID des Teilnehmers ist $7D_{16} = 125_{10}$.
Dann lautet Byte 1 der Bootup-Nachricht: $77D_{16} = 1\,917_{10}$

Es gibt Geräte, die kein [700_{16} + Node-ID] senden können.
Diese Geräte senden stattdessen folgende Bootup-Nachricht und ohne Status:

hex	80_{16} + Node-ID
dez	128_{10} + Node-ID

© ifm electronic gmbh

12.2.4 Netzwerk-Management (NMT)

9974

Netzwerk-Management-Kommandos

9962

Mit folgenden Netzwerk-Management-Kommandos kann der Anwender den Betriebsmodus von einzelnen oder allen CAN-Teilnehmern beeinflussen. Muster:

Byte 1	Byte 2	Byte 2
COB-ID	Kommando	Node-ID

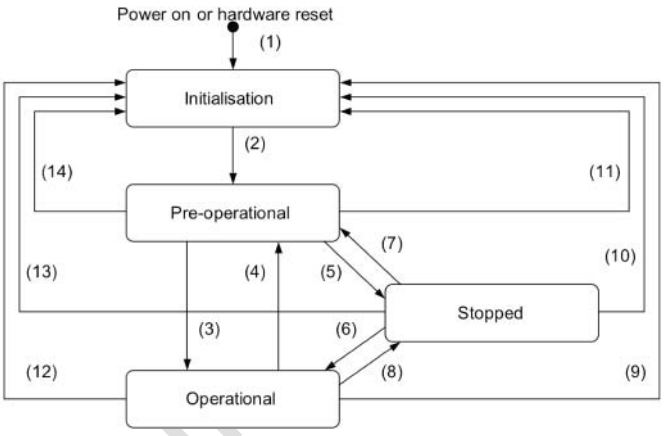
Node-ID = 00 ⇨ Kommando gilt zeitgleich für alle Knoten im Netz

COB-ID	NMT-Kommando		Beschreibung	
00	$01_{16} = 01_{10}$	Node-ID	start_remode_node	CAN-Teilnehmer starten
00	$02_{16} = 02_{10}$	Node-ID	stop_remode_node	CAN-Teilnehmer stoppen
00	$80_{16} = 128_{10}$	Node-ID	enter_pre-operational	umschalten auf Pre-Operational
00	$81_{16} = 129_{10}$	Node-ID	reset_node	CAN-Teilnehmer zurücksetzen
00	$82_{16} = 130_{10}$	Node-ID	reset_communication	CAN-Kommunikation zurücksetzen

NMT-Status

9963

Das Status-Byte gibt Auskunft über den Zustand des CAN-Teilnehmers.



Grafik: Zustandsübergänge unter CANopen

Erlaubte Übergänge:

(1) Zustand wird bei Power On automatisch erreicht.

(2) interne Initialisierung ist beendet – Knoten geht automatisch nach PRE-OPERATIONAL

(3) NMT Service "Start Remote Node Indication"

(4) + (7) NMT Service "Enter PRE-OPERATIONAL Indication"

(5) + (8) NMT Service "Stop Remote Node Indication"

(6) NMT Service "Start Remote Node Indication"

(9)...(11) NMT Service "Reset Node Indication"

(12)...(14) NMT Service "Reset Communication Indication"

© ifm electronic gmbh

NMT-Status für CANopen-Master

9964

Status hex	dez	Beschreibung
00	0	nicht definiert
01	1	Master wartet auf die Bootup-Nachricht des Slaves. ODER: Master wartet auf Ablauf der GuardTime.
02	2	- Master wartet 300 ms. - Master fordert das Objekt 1000_{16} an. - Danach wechselt der Master auf Status 3.
03	3	Der Master konfiguriert seine Slaves. Dazu sendet der Master an die Slaves der Reihe nach alle vom Konfigurator erzeugten SDOs: - Der Master sendet an den Slave ein SDO-Read-Request (Index 1000_{16}). - Die generierten SDOs werden in ein SDO-Array gepackt. - Der Slave kennt seine erste SDO und die Anzahl seiner SDOs.
05	5	Nachdem an alle Slaves die SDOs übertragen wurden, geht der Master in den Status 5 und bleibt in diesem Status. Status 5 ist für den Master der normale Betriebszustand.

Knoten-Status aus FB lesen:

verwendeter Funktionsblock	hier steht dieser Knoten-Status
CANx_MASTER_STATUS CANx_SLAVE_STATUS	Ausgang NODE_STATE
CANOPEN_GETSTATE	Ausgang NODESTATE

© ifm electronic gmbh

NMT-Status für CANopen-Slave

9965

Status hex | dez		Beschreibung
FF	-1	Der Slave wird durch die NMT-Nachricht [Reset Node] zurückgesetzt und wechselt selbständig in den Status 1.
00	0	nicht definiert
01	1	Status = Warten auf BOOTUP Der Slave wechselt nach einer maximalen Zeit von 2 s oder sofort nach Empfang seiner Bootup-Message in den Status 2.
02	2	Status = BOOTUP Der Slave wechselt nach einer Verzögerungszeit von 0,5 s automatisch in den Status 3.
03	3	Status = PREPARED Im Status 3 wird der Slave konfiguriert. Der Slave bleibt solange im Status 3, bis er alle vom Konfigurator erzeugten SDOs erhalten hat. Dabei spielt es keine Rolle, ob während der Konfiguration vom Slave SDO-Transfers mit Abort (Fehler) oder ob alle fehlerfrei beantwortet wurden. Nur die vom Slave erhaltene Antwort als solche ist wichtig – nicht ihr Inhalt. Wenn im Konfigurator die Option [Knoten zurücksetzen] aktiviert wurde, wird nach dem Senden des Objekts 1011_{16} Subindex 1, der dann den Wert "load" enthält, ein erneuter Reset des Slaves durchgeführt. Der Slave wird dann wieder mit dem Upload des Objekts 1000_{16} angefragt. Slaves, bei denen während der Konfigurationsphase ein Problem auftritt, bleiben im Status 3 oder wechseln nach der Konfigurationsphase direkt in einen Fehlerstatus (Status > 5).
04	4	Status = PRE-OPERATIONAL Ein Knoten wechselt immer in den Status 4, außer: • es handelt sich um einen "optionalen" Slave und er wurde als nicht am Bus verfügbar detektiert (Abfrage Objekt 1000_{16}) ODER: • der Slave ist zwar vorhanden, aber hat auf die Abfrage des Objekts 1000_{16} mit einem anderen Typ in den unteren 16 Bits reagiert, als der Konfigurator erwartet hat.
05	5	Status = OPERATIONAL Im Status 5 findet der normale Datenaustausch statt: "Normal Operation". Wenn der Master auf [Automatisch starten] konfiguriert wurde, wird der Slave im Status 4 gestartet (d.h. es wird eine "Start Node"-NMT-Nachricht erzeugt) und der Slave wechselt automatisch nach Status 5. Wurde GLOBAL_START gesetzt, dann wird gewartet, bis sich alle Slaves im Status 4 befinden. Anschließend werden alle Slaves mit dem NMT-Kommando [Start All Nodes] gestartet.
61	97	Ein Knoten wechselt in den Status 97, wenn er optional ist (optionales Gerät in der CAN-Konfiguration) und nicht auf die SDO-Anfrage nach dem Objekt 1000_{16} reagiert hat. Wird der Slave zu einem späteren Zeitpunkt an das Netzwerk angeschlossen und erkannt, wird er automatisch gestartet. Dazu müssen Sie aber die Option [Automatisch starten] in den CAN-Parametern des Masters angewählt haben.
62	98	Ein Knoten wechselt in den Status 98, wenn der Gerätetyp (Objekt 1000_{16}) nicht dem konfigurierten Typ entspricht.
63	99	Im Falle eines Nodeguarding-Timeouts wird der Slave auf Status 99 gesetzt. Sobald der Slave wieder auf NodeGuard-Anfragen reagiert und die Option [Automatisch starten] eingeschaltet ist, wird er automatisch vom Master gestartet. Dabei wird der Knoten abhängig von seinem Status, der in der Antwort auf die Nodeguard-Anfragen enthalten ist, neu konfiguriert oder nur gestartet. Um den Slave manuell zu starten, genügt es, die Methode [NodeStart] zu benutzen.

Der Master sendet Nodeguard-Nachrichten an den Slave, ...
- wenn sich der Slave im Status 4 oder höher befindet UND
- wenn Nodeguarding konfiguriert wurde.

© ifm electronic gmbh

Knoten-Status aus FB lesen:

verwendeter Funktionsblock	hier steht dieser Knoten-Status
CANx_MASTER_STATUS CANx_SLAVE_STATUS	Ausgang NODE_STATE
CANOPEN_GETSTATE	Ausgang NODESTATE

CANopen-Status des Knotens

1973

Knotenstatus nach CANopen (mit diesen Werten wird der Status auch in den entsprechenden Nachrichten vom Knoten her codiert).

Status hex | dez		CANopen-Status	Beschreibung
00	0	BOOTUP	Knoten hat die BOOTUP-Nachricht erhalten.
04	4	PREPARED	Knoten wird per SDOs konfiguriert.
05	5	OPERATIONAL	Knoten nimmt am normalen Datenaustausch teil.
7F	127	PRE-OPERATIONAL	Knoten sendet keine Daten, ist aber vom Master konfigurierbar.

Wenn Nodeguarding aktiv: das höchstwertige Status-Bit wechselt (toggelt) von Nachricht zu Nachricht.

Knoten-Status aus FB lesen:

verwendeter Funktionsblock	hier steht dieser Knoten-Status
CANx_MASTER_STATUS CANx_SLAVE_STATUS	Strukturelement LAST_STATE aus dem Array NODE_STATE_SLAVE
CANOPEN_GETSTATE	Ausgang LASTNODESTATE

© ifm electronic gmbh

12.2.5 CANopen Error-Code

Inhalt:

Emergency-Nachrichten 301
Übersicht CANopen Error-Codes 302
Objekt 0x1001 (Error-Register) 303

9967

Emergency-Nachrichten

9973

Gerätefehler im Slave oder Probleme im CAN-Bus lösen Emergency-Nachrichten aus:

COB-ID	DLC	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
80_{16} + Node-ID		Error-Code		Objekt 1001_{16}	gerätespezifisch				

Beachten Sie die umgekehrte Byte-Reihenfolge!

© ifm electronic gmbh

Übersicht CANopen Error-Codes

8545

Error Code (hex)	Meaning / Bedeutung
00xx	Reset or no Error (Fehler rücksetzen / kein Fehler)
10xx	Generic Error (allgemeiner Fehler)
20xx	Current (Stromfehler)
21xx	Current, device input side (Stromfehler, eingangsseitig)
22xx	Current inside the device (Stromfehler im Geräteinnern)
23xx	Current, device output side (Stromfehler, ausgangsseitig)
30xx	Voltage (Spannungsfehler)
31xx	Mains Voltage
32xx	Voltage inside the device (Spannungsfehler im Geräteinnern)
33xx	Output Voltage (Spannungsfehler, ausgangsseitig)
40xx	Temperature (Temperaturfehler)
41xx	Ambient Temperature (Umgebungstemperaturfehler)
42xx	Device Temperature (Gerätetemperaturfehler)
50xx	Device Hardware (Geräte-Hardware-Fehler)
60xx	Device Software (Geräte-Software-Fehler)
61xx	Internal Software (Firmware-Fehler)
62xx	User Software (Applications-Software)
63xx	Data Set (Daten-/Parameterfehler)
70xx	Additional Modules (zusätzliche Module)
80xx	Monitoring (Überwachung)
81xx	Communication (Kommunikation)
8110	CAN Overrun-objects lost (CAN Überlauf-Datenverlust)
8120	CAN in Error Passiv Mode (CAN im Modus "fehlerpassiv")
8130	Life Guard Error or Heartbeat Error (Guarding-Fehler oder Heartbeat-Fehler)
8140	Recovered from Bus off (Bus-Off zurückgesetzt)
8150	Transmit COB-ID collision (Senden "Kollision des COB-ID")
82xx	Protocol Error (Protokollfehler)
8210	PDO not procedded due to length error (PDO nicht verarbeitet, fehlerhafte Längenangabe)
8220	PDO length exceeded (PDO Längenfehler, ausgangsseitig)
90xx	External Error (Externer Fehler)
F0xx	Additional Functions (zusätzliche Funktionen)
FFxx	Device specific (gerätespezifisch)

© ifm electronic gmbh

Objekt 0x1001 (Error-Register)

8547

Dieses Objekt spiegelt den allgemeinen Fehlerzustand eines CANopen-Gerätes wider. Das Gerät ist dann als fehlerfrei anzusehen, wenn das Objekt 1001_{16} keinen Fehler mehr signalisiert.

Bit	Meaning (Bedeutung)
0	Generic Error (allgemeiner Fehler)
1	Current (Stromfehler)
2	Voltage (Spannungsfehler)
3	Temperature (Temperaturfehler)
4	Communication Error (Kommunikationsfehler)
5	Device Profile specific (Geräteprofil spezifisch)
6	Reserved – always 0 (reserviert – immer 0)
7	manufacturer specific (herstellerspezifisch)

Für eine Fehlermeldung können mehrere Bits im Error-Register gleichzeitig gesetzt sein.

Beispiel: CR2033, Meldung "Leitungsbruch" an Kanal 2 (→ Installationsanleitung des Geräts):

COB-ID	DLC	Byte 0	Byte 1	Byte	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4
80_{16} + Node-ID		**00**	**FF**	**81**	**10**	**00**	00	00	00

Error-Code = $FF00_{16}$

Error-Register = 81_{16} = $1000\ 0001_2$, besteht also aus folgenden Fehlern:
- Generic Error (allgemeiner Fehler)
- manufacturer specific (herstellerspezifisch)

Betroffener Kanal = 0010_{16} = $0000\ 0000\ 0001\ 0000_2$ = Kanal 2

© ifm electronic gmbh

12.3 Visualisierungen im Gerät

3111

In diesem Kapitel finden Sie wichtige Informationen über Bitmap-Grafiken in CoDeSys-Visualisierungen.

12.3.1 Grundsätzliches

10464

Grundsätzlich können Sie neben den grafischen Elementen, die Sie mit dem CoDeSys-Visualisierungs-Editor erstellen, auch Grafiken einbinden, die Sie mit anderen Programmen erstellt haben. Solche Grafikdateien können zum Beispiel Piktogramme, Logos oder auch kleine Bilder sein. Bevor Sie aber so eine "externe Grafik" einbinden, sind einige grundlegende Dinge zu beachten, die in den folgenden Kapiteln erläutert werden.

Weitere Hinweise finden Sie z.B. hier:

- Visualisierungen erstellen und parametrieren:
  → CoDeSys-Programmierhandbuch (→ **ecomat*mobile***-DVD "Software, tools and documentation")
  → **ifm**-Lehrbuch "PDM – Handbuch zur Einführung"
- Beachten Sie die *Begrenzungen und Programmierhinweise* (→ Seite 69)!

© ifm electronic gmbh

12.3.2 Empfehlungen für Bedienoberflächen

7435

Entscheidend für die Akzeptanz und den Gebrauch von technischen Produkten ist in hohem Maß ihre Benutzerfreundlichkeit!

In diesem Kapitel geben wir einige Empfehlungen, wie die Benutzeroberfläche (auch **H**uman-**M**achine-**I**nterface HMI genannt) einer Maschine möglichst nutzerfreundlich zu gestalten ist.

Empfehlungen zur nutzerfreundlichen Produktgestaltung

7436

Alle wichtigen Schnittstellen zwischen Mensch und Maschine werden durch Oberfläche und Gestaltung bestimmt. Wichtigen Kriterien für Gestaltung von Schnittstellen zwischen Mensch und Maschine sind…

- Eindeutigkeit:
  - Für jede Funktion eine eindeutige Funktionsbeschreibung.
  - Erwartungskonforme Gestaltung, Erlerntes bleibt gleich
- Ablesbarkeit:
  - Umgebung (Beleuchtung, Lese-Abstand) berücksichtigen.
- Intuitive Bedienbarkeit:
  - Stellteil / Funktion muss erkennbar sein.
  - Bedienoberfläche muss sich selbst erklären.
- Sinnlichkeit
  - Bedienelemente müssen nutzerfreundlich sein.
  - Gute Unterscheidbarkeit von anderen Anzeigen und Bedienelementen.
- Feedback
  - Zeitnahe Reaktion auf Nutzer-Aktivitäten.
  - Ursache für eine Meldung muss eindeutig erkennbar sein.
- Umgebung des Produkts wegen Ablenkung oder Irritation durch...
  - Lärm
  - Dunkelheit
  - Lichtreflexe
  - Vibrationen
  - extreme Temperaturen

Aus Sicht des Herstellers ist zusätzlich wichtig:

- Anzeige als markenspezifisches Merkmal.
- Anzeige muss Standards und Normen erfüllen.

© ifm electronic gmbh

Kennen Sie die künftigen Nutzer?

7444

Die künftigen Nutzer des Produkts sollten bekannt sein:

- Alter
- Geschlecht
- Sinne:
  - Sehfähigkeit
  - Höhrfähigkeit
  - bevorzugte Hand (Rechts- oder Linkshänder)
  - Tastfähigkeit
- Ausbildung:
  - allgemeines Ausbildungsniveau
  - spezifische Schulungen und Erfahrungen
- Motivation und kognitive Fähigkeiten:
  - Wahrnehmen (Sinnesorgane): Nicht alle zur Verfügung stehenden Informationen werden genutzt, sondern massiv gefiltert, integriert und auf viele andere Weisen verändert, bevor sie ins Bewusstsein gelangen.
  - Denken: Das Arbeitsgedächtnis, in dem die geistige Manipulation von Informationen stattfindet, hat eine sehr kleine Kapazität.
  - Lernen: Die im Langzeitgedächtnis gespeicherten Informationen werden häufig sowohl im Voraus (z.B. durch Erwartungen), als auch im Nachhinein (z.B. durch nachfolgende Informationen) verändert.
  - Erinnern: Die im Langzeitgedächtnis "eigentlich" vorhandenen Informationen sind häufig nicht abrufbar.
  - Motivation und Konzentration: Müdigkeit, Lustlosigkeit, Ablenkbarkeit usw. können die kognitive Leistungsfähigkeit beeinträchtigen.
- Vertrautheit mit dem Problem oder Anwendungsgebiet:
  - Gefahren erkennen können
  - Wissen, was nach einer Bedienung geschehen soll
- Intensität der Anwendung (wie oft und wie intensiv wird das Produkt benutzt)
- Kulturkreis, z.B.:
  - Sprache
  - Bedeutung von Farben und Symbolen
  - Leserichtung

© ifm electronic gmbh

Gebrauchstauglichkeit prüfen

7422

In vielen Fällen kann eine Versuchsanordnung mit potentiellen Nutzern wichtige Ergebnisse liefern, wo und wie das Produkt verbessert werden soll/muss, um am Markt erfolgreich zu sein.

Für desen sogenannten "Usability-Test" müssen nacheinander folgende Schritte durchlaufen werden:

- Benutzergruppe (Zielgruppe) feststellen:
  - Wer soll mit dem Produkt umgehen können?
- Interview-Leitfaden erstellen:
  - Mit welcher Methode befrage ich welche Nutzer (Bediener, Einrichter, Wartungspersonal)?
  - Was will ich mit den Interviews erreichen? (Verbesserungspotentiale)
- Interviews durchführen und auswerten.
- Kontext-Szenarien verfassen:
  - Auswertbare Prüfumgebung erstellen.
  - Kritische Nutzungs-Szenarien identifizieren.
- Nutzungstest durchführen:
  - Wie kommen die Prüfpersonen mit dem Produkt in der Versuchsanordnung zurecht?
  - Wo ergibt sich welcher Korrekturbedarf am Produkt?
- Nach erfolgter Optimierung des Produkts bei Bedarf die Tests wiederholen.

Sprache als Hindernis

7454

Um Geräte zu produzieren, die weltweit die Endkunden zufrieden stellen, muss die Sprache berücksichtigt werden. Der Bediener kann seine Aufgaben nicht effektiv erledigen, wenn er die Anweisungen auf dem Bildschirm nicht versteht. Hersteller versuchen immer noch, dieses Problem angesichts der vielen verschiedenen Sprachen weltweit zu lösen. Einige Sprachen sind nachstehend aufgeführt:

Chinesische Zeichen

Das chinesische Schriftzeichen, auch bekannt als Han-Chinesisch, ist ein Wortzeichen, d.h. es kann als Wort dargestellt werden. Die Anzahl der Zeichen in dem Kangxi-Wörterbuch liegt über 47 000, doch in China reicht es aus, wenn drei- bis viertausend Zeichen bekannt sind. In der Neuzeit sind die chinesischen Schriftzeichen sehr vereinfacht worden und werden in Festlandchina verwendet, während die traditionellen chinesischen Schriftzeichen noch in Hongkong und Taiwan verwendet werden. Die Chinesischen Zeichen sind romanisiert worden. Diese werden Pinyin genannt und sind in China auch weit verbreitet.

Japanische Schriftzeichen

Das moderne japanische Schriftsystem verwendet drei Hauptschriften:

- Kanji sind Ideogramme aus chinesischen Schriftzeichen
- Hiragana wird verwendet für muttersprachliche japanische Wörter und
- Katakana wird verwendet für Lehnwörter
- Romanisierte japanische Zeichen, Romanji genannt, werden ebenfalls in japanischen Texten verwendet.

© ifm electronic gmbh

Koreanische Schriftzeichen

Das moderne koreanische Schriftsystem wird Hangul genannt und offiziell in Nord- und Südkorea verwendet. Daneben wird Hanja verwendet, das sich auf die dem Chinesischen entlehnten Zeichen bezieht.

Arabisches Alphabet

Diese Schrift wird verwendet, um mehrere Sprachen in Asien (z.B. Mittlerer und Naher Osten, Pakistan) und Afrika (z.B. Arabisch und Urdu) zu schreiben. Sie ist eine Schreibschrift von rechts nach links und umfasst 28 Buchstaben.

Unicode

Unicode ist ein Standard für die konsequente Darstellung und Verwendung von Zeichen, die in den weltweiten Schriftsystemen vorkommen. Es war nicht leicht, Sprachen an Computer anzupassen, teilweise wegen der großen Anzahl von Zeichen in einigen Sprachen. Es ist möglich, ein englisches Zeichen mit nur einem Byte zu kodieren, weil Schriftenglisch nur wenige Zeichen benötigt. Das gilt nicht für Sprachen wie Japanisch, Chinesisch oder Koreanisch, die über 256 Zeichen haben und somit eine Doppel- oder Multibyte-Kodierung erfordern. Mehrere Kodierverfahren werden verwendet und Unicode scheint das universellste Verfahren zu sein. Es kodiert offensichtlich in alle Sprachen der Welt.

Zum Beispiel handelt es sich bei der Han-Vereinheitlichung, die zu Unihan zusammengezogen wird, um ein Unterfangen von Unicode und Universal Character Set (nach ISO 10646), mehrere Zeichensätze des Chinesischen, Japanischen und Koreanischen in einen einzigen Satz vereinheitlichter Zeichen abzubilden.

Arabische Schriftzeichen können kodiert werden durch Unicode ab Version 5.0 (mehrere Zeichensätze nach ISO 8859-6).

ISO 10646 spezifiziert den Universal Multiple Octet Coded Character Set. Er wird angewendet für Darstellung, Austausch, Verarbeitung, Speicherung und Eingabe der schriftlichen Form der weltweiten Sprachen sowie für zusätzliche Symbole.

Die Unicode-Standard-Versionen 4...6 entsprechen alle ISO 10646.

Piktogramm

Dies ist ein grafisches Symbol, auch Bildzeichen genannt, das ein Konzept, Objekt, Ereignis oder eine Aktivität durch Abbildung darstellt. Piktogramme gibt es seit vielen tausend Jahren. Sie spielen immer noch eine wichtige Rolle bei Sprachbarrieren und Analphabetismus in der modernen Welt und werden als Bildzeichen, Repräsentationszeichen, Anweisungen oder statistische Diagramme verwendet. Aufgrund ihrer grafischen Darstellung werden sie in unterschiedlichen Lebensbereichen eingesetzt. Um zum Beispiel auf Toiletten und Flughäfen hinzuweisen, wird ein Standardsatz von Piktogrammen definiert in der ISO 7001 "Graphische Symbole zur Information der Öffentlichkeit".

Ein Piktogramm ist zu einer funktionellen visuellen Sprache für Leute mit kognitiven Schwierigkeiten entwickelt worden. Jedes Bild steht für ein Wort oder ein Konzept. Es enthält zwei Elemente, gezeichnete Bilder und Text. Die Symbole sind meistens weiß auf einem schwarzen Quadrat.

© ifm electronic gmbh

Kulturelle Details sind oft nicht übertragbar

7461

Länder-, kultur- oder sprachspezifische Details sollten im Ausgangstext vermieden werden, da ihre Verwendung oft unnötig und ihre Anpassung an die Zielkultur sehr zeitraubend ist. Meistens weiß der Autor nicht, dass seine Texte oder Grafiken kulturell oder sprachlich geprägt sind oder dass sie durch andere gestalterische Entscheidungen Lokalisierungsprobleme erzeugen. Probleme können z.B. in folgenden Bereichen entstehen:

- Farben
- Symbole
- Abbildungen
- Leserichtung

Farben

7464

Die Wahl der "richtigen" Farbe ist ein wichtiges Element bei der Gestaltung von Text und Produkt. Viele Farben sind kulturspezifisch belegt und können bei falscher Verwendung zu Missverständnissen und über Fehlbedienungen sogar zum Imageverlust des Produkts führen.

Beispiele:

	Farbe	Bedeutung in Europa + USA	Bedeutung in anderen Kulturen
	rot	Dramatik, Umbruch, Blut (Kampf, Rache und Tod), Liebe, Gefahr, Adel	China: Glück, fröhlich Russland: schön Ägypten: Tod Indien: Leben, kreativ Japan: Ärger, Gefahr
	gelb	Vorsicht, Warnung, Sonnenlicht, Ewigkeit, Neid, Hass	China: Geburt, Gesundheit, Kraft Ägypten: fröhlich, Besitz Indien: Erfolg Japan: Adel
	grün	Natur, Ökologie, Hoffnung, unsterblich, Glück	China: Ewigkeit, Familie, Harmonie, Gesundheit, Frieden, die Nachwelt Agypten: fruchtbar, Stärke Indien: Besitz, fruchtbar Japan: Zukunft, Jugend, Energie
	blau	Wasser, Himmel, Treue, Freiheit, beständig, Freude, Freundschaft, männlich	Asien: Reichtum, Stärke Ägypten: Tugend, Glaube, Wahrheit
	weiß	Licht, rein, weise, Leben, vollkommen, ideal, gut, sachlich, klar, unschuldig, ehrlich	Asien: Tod, Trauer, Reinheit Ägypten: Freude
	schwarz	Tod, Trauer, Finsternis, das Böse. Auch: Brüderlichkeit, Macht und Einigkeit	(Trauer nicht im Buddhismus) Ägypten: Auferstehung
	grau	Weisheit und Alter	Asien: hilfreich

© ifm electronic gmbh

Symbole

7465

Da Symbole oft in Analogie zu kulturspezifischen Konzepten entstehen oder Anspielungen auf vertraute Bereiche der Ausgangskultur nutzen, stellen sie ein Problem für die Lokalisierung dar.

Beispiel:

Das Symbol für ein Haus, das für Start oder Anfang stehen soll, ist nicht eindeutig verständlich, da sich die englische Benennung "home" nicht problemlos übertragen lässt.

Abbildungen

7466

Nicht immer kann ein Bild einen Text sinnvoll ersetzen.

Die Darstellung komplexerer Prozesse kann unmöglich werden. Denn wie soll z.B. die Abbildung für die Aufforderung aussehen "Drücken Sie die Taste, bis Sie einen leichten Widerstand spüren"?

Und selbst wenn eine Abbildung einen Sachverhalt gut darstellen kann, muss ihr Einsatz auf internationaler Ebene gut durchdacht werden. Das Ersetzen von Text durch Bilder ist nämlich nur dann sinnvoll und kostensenkend, wenn die Abbildungen kulturneutral, also in ALLEN angestrebten Zielländern ohne Anpassungen einsetzbar sind. Viele Dinge, die uns hier völlig selbstverständlich erscheinen, sind es in anderen Kulturen nicht.

Die Abbildung von Menschen kann zu Problemen führen: Welches Geschlecht soll oder darf die Person haben? Welche Hautfarbe? Welches Alter? Schließlich sollen sich die Adressaten in allen Zielländern gleichermaßen angesprochen fühlen. Kleidung, die in Westeuropa unauffällig ist, kann in arabischen oder afrikanischen Ländern zu Irritationen führen. Auch die Darstellung von Gesten und einzelnen Körperteilen, speziell von Händen und Augen, sollte unterbleiben, da diese oft eine anstößige oder beleidigende Assoziation auslösen.

Leserichtung

7468

In den meisten Kulturen wird von links nach rechts und von oben nach unten gelesen.

Einige asiatische Kulturen lesen jedoch von unten nach oben und von hinten nach vorn.

Viele arabische Kulturen lesen von rechts nach links.

Diese Besonderheiten sind auch bei rein grafischen Anleitungen zu beachten!

© ifm electronic gmbh

Richtlinien und Normen

7445

Die folgende Aufstellung ist nur eine Auswahl und erhebt keinen Anspruch auf Vollständigkeit.

ISO 7001 _ Graphische Symbole zur Information der Öffentlichkeit

7456

Ein grafisches Symbol, auch Bildzeichen genannt, stellt ein Konzept, Objekt, Ereignis oder eine Aktivität durch Abbildung dar. Piktogramme gibt es seit vielen tausend Jahren. Sie spielen immer noch eine wichtige Rolle bei Sprachbarrieren und Analphabetismus in der modernen Welt und werden als Bildzeichen, Repräsentationszeichen, Anweisungen oder statistische Diagramme verwendet. Aufgrund ihrer grafischen Darstellung werden sie in unterschiedlichen Lebensbereichen eingesetzt.

Beispiele:

© ifm electronic gmbh

ISO 9126 _ Qualitätsmerkmale für Software-Produkte

7446

Die Norm beschreibt folgende Kriterien:

Funktionalität: Inwieweit besitzt die Software die geforderten Funktionen?

- Angemessenheit: Eignung von Funktionen für spezifizierte Aufgaben, z. B. aufgabenorientierte Zusammensetzung von Funktionen aus Teilfunktionen.
- Richtigkeit: Liefern der richtigen oder vereinbarten Ergebnisse oder Wirkungen, z. B. die benötigte Genauigkeit von berechneten Werten.
- Interoperabilität: Fähigkeit, mit vorgegebenen Systemen zusammenzuwirken.
- Sicherheit: Fähigkeit, unberechtigten Zugriff (versehentlich oder vorsätzlich) auf Programme und Daten zu verhindern.
- Ordnungsmäßigkeit: Merkmale von Software, die bewirken, dass die Software anwendungsspezifische Normen oder Vereinbarungen oder gesetzliche Bestimmungen und ähnliche Vorschriften erfüllt.

Zuverlässigkeit: Kann die Software ein bestimmtes Leistungsniveau unter bestimmten Bedingungen über einen bestimmten Zeitraum aufrechterhalten?

- Reife: Geringe Versagenshäufigkeit durch Fehlerzustände.
- Fehlertoleranz: Fähigkeit, ein spezifiziertes Leistungsniveau bei Software-Fehlern oder Nicht-Einhaltung ihrer spezifizierten Schnittstelle zu bewahren.
- Robustheit: Fähigkeit, ein stabiles System bei Eingaben zu gewährleisten, die nicht vorgesehen sind. Die Software hält "DAUs" stand.
- Wiederherstellbarkeit: Fähigkeit, bei einem Versagen das Leistungsniveau wiederherzustellen und die direkt betroffenen Daten wiederzugewinnen. Zu berücksichtigen sind die dafür benötigte Zeit und der benötigte Aufwand.
- Konformität: Grad, in dem die Software Normen oder Vereinbarungen zur Zuverlässigkeit erfüllt.

Benutzbarkeit: Welchen Aufwand fordert der Einsatz der Software von den Benutzern und wie wird er von diesen beurteilt?

- Verständlichkeit: Aufwand für den Benutzer, das Konzept und die Anwendung zu verstehen.
- Erlernbarkeit: Aufwand für den Benutzer, die Anwendung zu erlernen (z. B. Bedienung, Ein-, Ausgabe).
- Bedienbarkeit: Aufwand für den Benutzer, die Anwendung zu bedienen.
- Attraktivität: Anziehungskraft der Anwendung gegenüber dem Benutzer.
- Konformität: Grad, in dem die Software Normen oder Vereinbarungen zur Benutzbarkeit erfüllt.

Effizienz: Wie liegt das Verhältnis zwischen Leistungsniveau der Software und eingesetzten Betriebsmitteln?

- Zeitverhalten: Antwort- und Verarbeitungszeiten sowie Durchsatz bei der Funktionsausführung.
- Verbrauchsverhalten: Anzahl und Dauer der benötigten Betriebsmittel bei der Erfüllung der Funktionen. Ressourcenverbrauch, wie CPU-Zeit, Festplattenzugriffe usw.
- Konformität: Grad, in dem die Software Normen oder Vereinbarungen zur Effizienz erfüllt.

Änderbarkeit: Welchen Aufwand erfordert die Durchführung vorgegebener Änderungen an der Software? Änderungen können Korrekturen, Verbesserungen oder Anpassungen an Änderungen der Umgebung, der Anforderungen oder der funktionalen Spezifikationen einschließen.

- Analysierbarkeit: Aufwand, um Mängel oder Ursachen von Versagen zu diagnostizieren oder um änderungsbedürftige Teile zu bestimmen.
- Modifizierbarkeit: Aufwand zur Ausführung von Verbesserungen, zur Fehlerbeseitigung oder Anpassung an Umgebungsänderungen.
- Stabilität: Wahrscheinlichkeit des Auftretens unerwarteter Wirkungen von Änderungen.
- Testbarkeit: Aufwand, der zur Prüfung der geänderten Software notwendig ist.

© ifm electronic gmbh

Übertragbarkeit: Wie leicht lässt sich die Software in eine andere Umgebung übertragen? Umgebung kann organisatorische Umgebung, Hardware- oder Software-Umgebung sein.

- Anpassbarkeit: Fähigkeit der Software, diese an verschiedene Umgebungen anzupassen.
- Installierbarkeit: Aufwand, der zum Installieren der Software in einer festgelegten Umgebung notwendig ist.
- Koexistenz: Fähigkeit der Software neben einer anderen mit ähnlichen oder gleichen Funktionen zu arbeiten.
- Austauschbarkeit: Möglichkeit, diese Software anstelle einer spezifizierten anderen in der Umgebung jener Software zu verwenden, sowie der dafür notwendige Aufwand.
- Konformität: Grad, in dem die Software Normen oder Vereinbarungen zur Übertragbarkeit erfüllt.

© ifm electronic gmbh

ISO 9241 _ Ergonomie der Mensch-System-Interaktion

7447

Die Norm ISO 9241 ist ein internationaler Standard, der Richtlinien der Interaktion zwischen Mensch und Computer beschreibt. Die Normenreihe beschreibt Anforderungen an die Arbeitsumgebung, Hardware und Software. Ziel der Richtlinie ist es, gesundheitliche Schäden beim Arbeiten am Bildschirm zu vermeiden und dem Benutzer die Ausführung seiner Aufgaben zu erleichtern.

Die folgenden Teile (jedoch nicht ausschließlich) sind Bestandteile der Norm:

Teil 1: Allgemeine Einführung

Teil 2: Anforderungen an die Arbeitsaufgaben – Leitsätze

Teil 3: Anforderungen an visuelle Anzeigen

Teil 4: Anforderungen an Tastaturen

Teil 5: Anforderungen an die Arbeitsplatzgestaltung und Körperhaltung

Teil 6: Anforderungen an die Arbeitsumgebung

Teil 7: Anforderungen an visuelle Anzeigen bezüglich Reflexionen

Teil 8: Anforderungen an Farbdarstellungen

Teil 9: Anforderungen an Eingabegeräte - außer Tastaturen

(Teil 10: Grundsätze der Dialoggestaltung (veraltet, da seit 2006 ersetzt durch Teil 110))

Teil 11: Anforderungen an die Gebrauchstauglichkeit – Leitsätze

Teil 12: Informationsdarstellung

Teil 13: Benutzerführung

Teil 14: Dialogführung mittels Menüs

Teil 15: Dialogführung mittels Kommandosprachen

Teil 16: Dialogführung mittels direkter Manipulation

Teil 17: Dialogführung mittels Bildschirmformularen

Teil 110: Grundsätze der Dialoggestaltung (ersetzt den bisherigen Teil 10)

Teil 151: Leitlinien zur Gestaltung von Benutzungsschnittstellen für das World Wide Web

Teil 171: Leitlinien für die Zugänglichkeit von Software (im Oktober 2008 veröffentlicht)

Teil 300: Einführung in Anforderungen und Messtechniken für elektronische optische Anzeigen

Teil 302: Terminologie für elektronische optische Anzeigen (zurzeit im Entwurfsstadium)

Teil 303: Anforderungen an elektronische optische Anzeigen (zurzeit im Entwurfsstadium)

Teil 304: Prüfverfahren zur Benutzerleistung

Teil 305: Optische Laborprüfverfahren für elektronische optische Anzeigen (zurzeit im Entwurfsstadium)

Teil 306: Vor-Ort-Bewertungsverfahren für elektronische optische Anzeigen (zurzeit im Entwurfsstadium)

Teil 307: Analyse und Konformitätsverfahren für elektronische optische Anzeigen (zurzeit im Entwurfsstadium)

Teil 400: Grundsätze und Anforderungen für physikalische Eingabegeräte

Teil 410: Gestaltungskriterien für physikalische Eingabegeräte (zurzeit im Entwurfsstadium)

Die Teile 5 und 6 umfassen den Themenbereich Arbeitsumgebung. Die Teile 3, 4, 7, 8 und 9 beschäftigen sich mit Anforderungen an Hardware, während die Teile 11...17 und 110 Aspekte der Software-Ergonomie behandeln. Vor allem in den Teilen *ISO 9241-110 _ Grundsätze der Dialoggestaltung* (→ Seite 315) und *ISO 9241-11 _ Anforderungen an die Gebrauchstauglichkeit* (→ Seite 315) finden sich einige Kriterien für die ergonomische Gestaltung interaktiver Systeme.

© ifm electronic gmbh

ISO 9241-11 _ Anforderungen an die Gebrauchstauglichkeit

7448

Die Gebrauchstauglichkeit einer Software ist von ihrem Nutzungskontext abhängig. Im Teil 11 der ISO 9241 werden drei Leitkriterien für die Gebrauchstauglichkeit einer Software bestimmt:

- Effektivität zur Lösung einer Aufgabe,
- Effizienz der Handhabung des Systems,
- Zufriedenheit der Nutzer einer Software.

ISO 9241-110 _ Grundsätze der Dialoggestaltung

7450

Benutzungsschnittstellen von interaktiven Systemen, wie Webseiten oder Software, sollten vom Benutzer leicht zu bedienen sein. Der Teil 110 der ISO 9241 beschreibt folgende Grundsätze für die Gestaltung und Bewertung einer Schnittstelle zwischen Benutzer und System (Dialoggestaltung):

- Aufgabenangemessenheit
  geeignete Funktionalität, Minimierung unnötiger Interaktionen
- Selbstbeschreibungsfähigkeit
  Verständlichkeit durch Hilfen / Rückmeldungen
- Lernförderlichkeit
  Anleitung des Benutzers, Verwendung geeigneter Metaphern, Ziel: minimale Erlernzeit
- Steuerbarkeit
  Steuerung des Dialogs durch den Benutzer
- Erwartungskonformität
  Konsistenz, Anpassung an das Benutzermodell
- Individualisierbarkeit
  Anpassbarkeit an Benutzer und an seinen Arbeitskontext
- Fehlertoleranz
  Intelligente Dialoggestaltung zur Fehlervermeidung seitens der Benutzer steht an erster Stelle.
  Ansonsten: erkannte Fehler des Benutzers verhindern nicht das Benutzerziel.
  Unerkannte Fehler: leichte Korrektur durch den Benutzer.

© ifm electronic gmbh

ISO 10646 _ Informationstechnik – Universeller Mehrfach-8-bit-codierter Zeichensatz (UCS)

7455

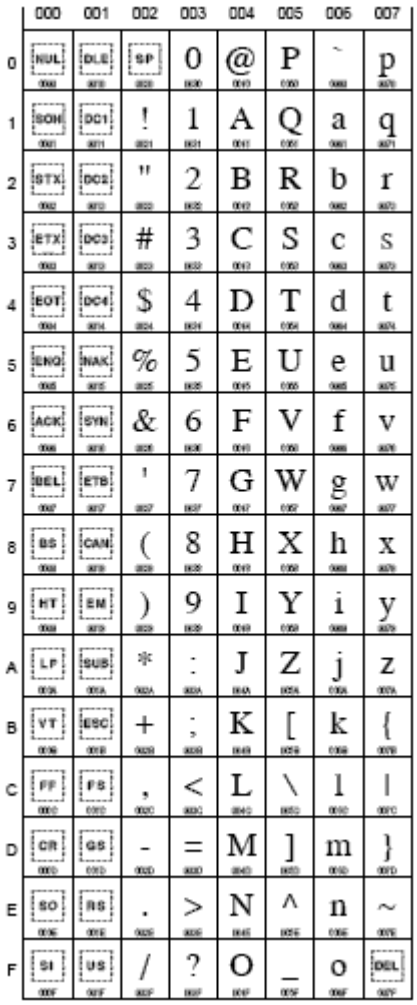
Der Universal Character Set (UCS) ist eine Zeichenkodierung, die im internationalen Standard ISO 10646 definiert ist. Für alle praktischen Belange ist dies dasselbe wie Unicode.

Pro Zeichen werden 2 Byte Speicherplatz verwendet. Entsprechend ist Unicode ein 16-Bit-Code, mit dem man 2^{16} = 65 536 Zeichen repräsentieren kann. Erstes Ziel ist, die Schriftzeichen aller Staatssprachen eindeutig und einheitlich zu kodieren.

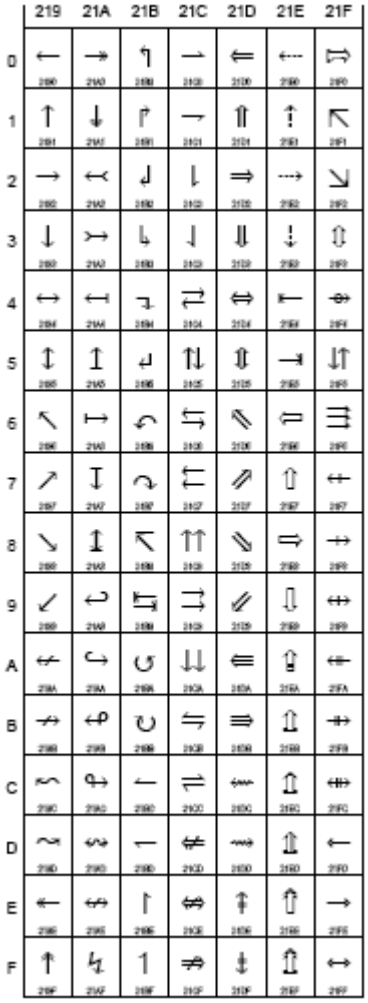
Nicht alle dieser 65 536 Zeichenadressen werden dabei standardisiert belegt: Ein nutzerdefinierter Bereich erlaubt es, ca. 2 000 Adressen mit nutzerspezifischen Zeichen zu belegen.

Über die Kombination von zwei 16-Bit-Codes kann man weitere 1 408 576 Zeichen ansprechen. Damit hofft man, alle Schriftzeichen erfassen zu können, die es gibt und jemals gegeben hat. Darüberhinaus werden auch technische Symbole, musikalische Zeichen, Lautschrift etc. abgebildet. Noch sind aber bei weitem nicht alle Zeichenadressen belegt.

Beispiele:


000 001 002 003 004 005 006 007
0 NUL DLE SP 0 @ P ` p
1 SOH DC1 ! 1 A Q a q
2 STX DC2 " 2 B R b r
3 ETX DC3 # 3 C S c s
4 EOT DC4 $ 4 D T d t
5 ENQ NAK % 5 E U e u
6 ACK SYN & 6 F V f v
7 BEL ETB ' 7 G W g w
8 BS CAN (8 H X h x
9 HT EM) 9 I Y i y
A LF SUB * : J Z j z
B VT ESC + ; K [k {
C FF FS , < L \ l |
D CR GS - = M] m }
E SO RS . > N ^ n ~
F SI US / ? O _ o DEL

Unicode: Steuerzeichen und Basiszeichen



Unicode: Pfeile

ISO 13406 _ Ergonomische Anforderungen für Tätigkeiten an optischen Anzeigeeinheiten in Flachbauweise

7453

Teil 2: Ergonomische Anforderungen an Flachbildschirme

Gemäß der internationalen Norm ISO 13406-2 werden LCD-Bildschirme nach folgenden Kriterien klassifiziert:

- Leuchtdichte, Kontrast und Farbe gemessen an der Blickrichtung des Betrachters
- Reflexionen und Kontrast bei einfallender Beleuchtung
- Bildaufbauzeit
- Defekte (Pixelfehler)

ISO 13407 _ Benutzer-orientierte Gestaltung interaktiver Systeme

7452

Die ISO 13407 ist eine Norm, die einen prototypischen benutzerorientierten Softwareentwicklungsprozess beschreibt. Ein spezieller Entwicklungsprozess kann als zu ihr konform betrachtet werden, wenn ihre Empfehlungen erfüllt werden.

Die Norm stellt nutzerorientierte Gestaltung als eine fachübergreifende Aktivität dar, die Wissen über menschliche Faktoren und ergonomische Kenntnisse und Techniken umfasst. Der ISO-Prozess besteht aus vier wesentlichen Teilaktivitäten:

- Nutzungskontext verstehen:
  Das Ergebnis dieser Aktivität ist eine dokumentierte Beschreibung der relevanten Benutzer, ihrer Arbeitsaufgaben und ihrer Umgebung.
- Anforderungen spezifizieren:
  Während dieser Phase werden die Zielgrößen aus der bereits vorhandenen Dokumentation auf einer Kompromissebene abgeleitet. Dabei wird die Teilung der Systemaufgaben bestimmt in...
  - solche, die von Menschen durchgeführt werden sollen
  - solche, die von der Technik durchgeführt werden sollen.
- Lösungen produzieren:
  Dies kann im Sinne einer Prototyp-Entwicklung oder eines anderen iterativen Prozesses erfolgen. Diese Prototypen können noch reine Papierentwürfe (Attrappen) oder aber schon lauffähige Programmversionen sein. Falls es unternehmensinterne Gestaltungsregeln für Benutzerschnittstellen gibt, sollten diese genutzt werden.
- Lösungen bewerten:
  Die Lösungen werden auf die Erfüllung der festgelegten Anforderungen geprüft. Dazu können Experten-Bewertungen, Gebrauchstauglichkeitstests (Usability-Tests), Befragungen oder auch eine Mischung daraus dienen. Die dabei entdeckten Abweichungen werden dann auf ihre Relevanz hin bewertet und sind Ausgangspunkt der nächsten Iteration des Entwicklungsprozesses.

Dieses Verfahren ist komplementär zu bestehenden Prozessmodellen der Software-Entwicklung und ergänzt diese. Der benutzerorientierte Gestaltungsprozess sollte der Norm zufolge bereits im frühesten Stadium des Projekts beginnen und sollte dann wiederholt durchlaufen werden, bis das System die Anforderungen erfüllt. Die Bedeutung und der Aufwand für die benutzerorientierte Gestaltung misst sich an der Größe und Art des zu entwickelnden Produkts und wird für kleinere Projekte durch Einzelpersonen gesteuert.

© ifm electronic gmbh

ISO 20282 _ Bedienungsfreundlichkeit von Produkten des täglichen Gebrauchs

7443

Dieser Normenentwurf besteht aus:

- Teil 1: Gebrauchsumfeld und Benutzerkriterien
  Beschreibt folgende Kriterien:
  - den Anwendungsbereich
  - die Benutzerschnittstelle
  - den Nutzer
  - seine psychischen und sozialen Charakteristika
  - die physische und soziale Umgebung
  - die physische und sensorische Kategorie.
- Teil 2: Prüfverfahren für öffentlich zugängliche Produkte
  Definiert als Technische Spezifikation die Prüfverfahren.

© ifm electronic gmbh

12.3.3 Grundlegende Informationen zu Bitmap-Grafiken

9858
3112

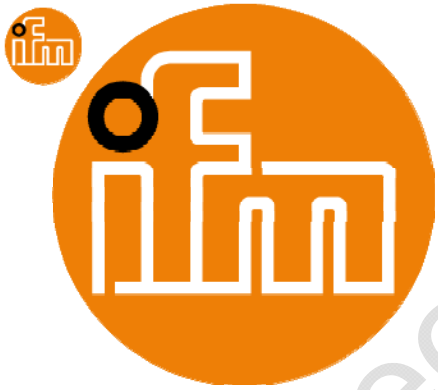
Bei Grafiken und Bilddateien unterscheidet man vereinfacht zwei grundsätzliche Typen:

	Vektorgrafiken	**Pixelgrafiken**
Beispiele:	Zeichnungen von CAD-Programmen. Zeichensätze vom Typ TrueType, PostScript oder OpenType.	Digitalfotos. Dateien aus dem Scanner oder aus Capture-Programmen.
Prinzip:	Vektorgrafiken basieren auf einer Bildbeschreibung, die die Objekte, aus denen das Bild aufgebaut ist, exakt definiert. Z.B. ein Kreis wird definiert über Lage (Koordinaten) des Mittelpunktes, Radius, Linienstärke und Farbe.	Eine Rastergrafik, auch Pixelgrafik oder Bitmap, ist eine Form der Beschreibung eines Bildes, bestehend aus einer rasterförmigen Anordnung von so genannten Pixeln (Bildpunkten), denen jeweils eine Farbe zugeordnet ist. Die Hauptmerkmale einer Rastergrafik sind daher die Breite und die Höhe in Pixeln (Bildauflösung) sowie die Farbtiefe.
Speicherbedarf:	Speicherbedarf relativ gering.	Je nach Auflösung ist Speicherbedarf hoch bis sehr hoch: Die Dateien werden mit jedem zusätzlich zu speichernden Bildpunkt immer größer.
Verluste beim Skalieren:	Verlustfreie Umrechnung (skalieren) in beliebige Bildgrößen möglich.	Umrechnung (skalieren) in andere Bildgrößen meist nur mit Qualitätsverlusten möglich.
Leistungsfähigkeit der Hardware:	Da Monitore grundsätzlich auf einer Raster-Matrix basieren, müssen alle Grafiken in einzelne Bildpunkte umgerechnet (= gerastert) werden, um sie auf dem Monitor anzeigen zu können. Je nach Komplexität der Grafik sehr leistungsfähige Rechner erforderlich, um eine schnelle Bearbeitung und Anzeige zu ermöglichen.	Anforderung relativ gering.
Typische Datei-Endungen:	*.cdr (Corel Draw) *.dwg (AutoCAD) *.ai (Adobe Illustrator) *.svg (Scalable Vector Graphics)	*.bmp (Bitmap) *.gif (Compuserv GIF) *.jpg (Joint Photographic Experts Group) *.png (Portable Network Graphics)

© ifm electronic gmbh

Bildgröße Vektorgrafik / Pixelgrafik

7380

Vektorgrafiken	Pixelgrafiken
Grafische Elemente werden als Vektoren beschrieben: Informationen über Start- und Endpunkt, Dicke und Farbe einer Linie, ggf. Füllmuster und Farbverlauf.	Pixelgrafiken aus modernen Digitalkameras haben 5 Millionen und mehr Bildpunkte (Auflösung = 5 Megapixel). Spezielle Datenkompression versucht, den hohen Speicherbedarf zu mindern. Leider arbeitet die Kompression nur mit Qualitätsverlust.
Vergrößern oder Verkleinern erfolgt einfach und ohne Qualitätsverluste (→ Beispiel unten).	Beim Vergrößern entstehen entweder Klötzchen-Grafiken oder verschwommene Bilder (→ Beispiel unten). Einen hohen Verlust an Bildinformationen hat man beim Verkleinern eines solchen Megapixel-Bildes.
Beispiel:  Original Ø 10 mm / Vergrößerung 5-fach EPS-Datei jeweils 35 kB	Beispiel: Original 30x30 px / Vergrößerung 5-fach BMP-Datei 3 kB / 62 kB

Beispiel: Verkleinern eines Pixelbildes

3120

Ein Digitalfoto mit einer Auflösung von 5 Megapixeln hat eine Bildgröße von 2 560 x 1 920 Bildpunkten (= 4 915 200 Pixeln). Dieses Foto soll nun in einer Bildgröße von nur 320 x 240 Bildpunkten (= Monitorgröße beim PDM) dargestellt werden.

Folge nach dem Skalieren: Es sind nur noch 76 800 Bildpunkte (= 1,56 % des ursprünglichen Bildes) verblieben, die anderen 4 838 400 Pixel entfallen ersatzlos.

Oder anders ausgedrückt: Senkrecht wie waagerecht wird nur jedes 8. Pixel verwendet.

Daher kann ein so gewandeltes Foto nicht mehr die Qualität des Originals besitzen. Wichtige Informationen gehen verloren.

► **Abhilfe:** Bilder von Anfang an in der benötigten Größe und Auflösung anfertigen.

© ifm electronic gmbh

Bitmap-Grafiken anpassen

9996

Vorhandene Bitmap-Grafiken können Sie mit gängigen Grafikprogrammen anpassen.
Fragen Sie Ihren **ecomat*mobile***-Fachberater!

© ifm electronic gmbh

12.4 Übersicht der verwendeten Dateien und Bibliotheken

Inhalt:

Dateien und Bibliotheken im Gerät installieren 322
Allgemeine Übersicht 323
Wozu dienen die einzelnen Dateien und Bibliotheken? 325

2711

(Stand: 02.03.2011)

Je nach Gerät und gewünschter Funktion kommen verschiedene Bibliotheken und Dateien zum Einsatz. Teilweise werden sie automatisch geladen oder müssen vom Programmierer eingefügt oder geladen werden.

12.4.1 Dateien und Bibliotheken im Gerät installieren

2721

Werkseinstellung: Das Gerät enthält nur den Bootloader.

► Betriebssystem (`*.H86` oder `*.RESX`) laden.

► Projekt (`*.PRO`) im PC anlegen: Target (`*.TRG`) eintragen.

► Zusätzlich je nach Gerät und Target-Version:
Steuerungskonfiguration (`*.CFG`) festlegen.

> CoDeSys bindet die zum Target zugehörenden Dateien in das Projekt ein:
`*.TRG, *.CFG, *.CHM, *.INI, *.LIB`.

► Bei Bedarf das Projekt mit weiteren Bibliotheken (`*.LIB`) ergänzen.

Bestimmte Bibliotheken binden automatisch weitere Bibliotheken in das Projekt ein:
z.B. basieren einige Funktionsblöcke in **ifm**-Bibliotheken (`ifm_*.LIB`) auf Funktionsblöcken in CoDeSys-Bibliotheken (`3S_*.LIB`).

© ifm electronic gmbh

12.4.2 Allgemeine Übersicht

2712

Dateiname	Beschreibung und Speicherort [3])
`ifm_CRnnnn_Vxxyyzz.CFG` [1]) `ifm_CRnnnn_Vxx.CFG` [2])	Steuerungskonfiguration je Gerät nur 1 gerätespezifische Datei enthält: IEC- und symbolische Adressen der Ein- und Ausgänge, der Systemmerker sowie die Speicherverteilung `…\CoDeSys V*\Targets\ifm\ifm_CRnnnncfg\Vxxyyzz`
`CAA-*.CHM`	Online-Hilfe je Gerät nur 1 gerätespezifische Datei enthält: Online-Hilfe zu diesem Gerät `…\CoDeSys V*\Targets\ifm\Help\…` (Sprache)
`ifm_CRnnnn_Vxxyyzz.H86` `ifm_CRnnnn_Vxxyyzz.RESX`	Betriebssystem / Laufzeitsystem (muss bei Erstbenutzung in den Controller / Monitor geladen werden) je Gerät nur 1 gerätespezifische Datei `…\CoDeSys V*\Targets\ifm\Library\ifm_CRnnnn`
`ifm_Browser_CRnnnn.INI`	CoDeSys-Bowser-Kommandos (CoDeSys benötigt die Datei zum Projektstart) je Gerät nur 1 gerätespezifische Datei enthält: Kommandos für Browser in CoDeSys `…\CoDeSys V*\Targets\ifm`
`ifm_Errors_CRnnnn.INI`	CoDeSys-Fehler-Datei (CoDeSys benötigt die Datei zum Projektstart) je Gerät nur 1 gerätespezifische Datei enthält: gerätespezifische Fehlermeldungen aus CoDeSys `…\CoDeSys V*\Targets\ifm`
`ifm_CRnnnn_Vxx.TRG`	Target-Datei je Gerät nur 1 gerätespezifische Datei enthält: Hardware-Beschreibung für CoDeSys, z.B.: Speicher, Dateiablageorte `…\CoDeSys V*\Targets\ifm`
`ifm_*_Vxxyyzz.LIB`	allgemeine Bibliotheken je Gerät mehrere Dateien möglich `…\CoDeSys V*\Targets\ifm\Library`
`ifm_CRnnnn_Vxxyyzz.LIB`	gerätespezifische Bibliothek je Gerät nur 1 gerätespezifische Datei enthält: Programmbausteine dieses Geräts `…\CoDeSys V*\Targets\ifm\Library\ifm_CRnnnn`
`ifm_CRnnnn_*_Vxxyyzz.LIB`	gerätespezifische Bibliotheken je Gerät mehrere Dateien möglich → folgende Tabellen `…\CoDeSys V*\Targets\ifm\Library\ifm_CRnnnn`

Legende:

*	beliebige Zeichen
CRnnnn	Artikelnummer des Controllers / Monitors
V*	CoDeSys-Version
Vxx	Versionsnummer der ifm-Software
yy	Release-Nummer der ifm-Software
zz	Patch-Nummer der ifm-Software

[1]) gültig für CRnn32 Target-Version bis V01, alle anderen Geräte bis V04

[2]) gültig für CRnn32 Target-Version ab V02, CR040n ab V01, alle anderen Geräte ab V05

[3]) Speicherort der Dateien:
`System-Laufwerk (C: / D:) \ Programme-Ordner \ ifm electronic`

© ifm electronic gmbh

HINWEIS

Es müssen immer die zum gewählten Target passenden Software-Stände zum Einsatz kommen:
- des Betriebssystems (`ifm_CRnnnn_Vxxyyzz.H86` / `ifm_CRnnnn_Vxxyyzz.RESX`),
- der Steuerungskonfiguration (`ifm_CRnnnn_Vxx.CFG`),
- der Gerätebibliothek (`ifm_CRnnnn_Vxxyyzz.LIB`) und
- der weiteren Dateien (→ Kapitel *Übersicht der verwendeten Dateien und Bibliotheken*, → Seite 322).

CRnnnn	Geräte-Artikelnummer
Vxx: 00...99	Versionsnummer
yy: 00...99	Release-Nummer
zz: 00...99	Patch-Nummer

Dabei müssen der Basisdateiname (z.B. "`CR0032`") und die Software-Versionsnummer "xx" (z.B. "`02`") überall den gleichen Wert haben! Andernfalls geht das Gerät in den STOP-Zustand

Die Werte für "yy" (Release-Nummer) und "zz" (Patch-Nummer) müssen **nicht** übereinstimmen.

Folgende Dateien müssen ebenfalls geladen sein:
- die zum Projekt erforderlichen internen Bibliotheken (in IEC 1131 erstellt),
- die Konfigurationsdateien (`*.CFG`)
- und die Target-Dateien (`*.TRG`).

Es kann vorkommen, dass das Zielsystem mit Ihrer aktuell installierten Version von CoDeSys nicht oder nur teilweise programmiert werden kann. Im diesem Fall wenden Sie sich bitte an den technischen Support der **ifm electronic gmbh**.

© ifm electronic gmbh

12.4.3 Wozu dienen die einzelnen Dateien und Bibliotheken?

Inhalt:

Dateien für Betriebssystem / Laufzeitsystem 325
Target-Datei 325
Steuerungskonfigurations-Datei 325
ifm-Gerätebibliotheken 326
ifm-CANopen-Hilfsbibliotheken Master/Slave 326
CoDeSys-CANopen-Bibliotheken 327
spezielle ifm-Bibliotheken 328

2713

Die nachfolgende Übersicht zeigt, welche Dateien/Bibliotheken mit welchem Gerät eingesetzt werden können und dürfen. Dateien/Bibliotheken, die in dieser Liste nicht aufgeführt werden, können nur unter bestimmten Bedingungen eingesetzt werden oder die Funktionalität wurde noch nicht getestet.

Dateien für Betriebssystem / Laufzeitsystem

2714

Dateiname	Funktion	verfügbar für:
`ifm_CRnnnn_Vxxyyzz.H86` `ifm_CRnnnn_Vxxyyzz.HEX`	Betriebssystem / Laufzeitsystem	alle ecomatmobile-Controller BasicDisplay: CR0451 PDM: CR10nn
`ifm_Browser_CRnnnn.INI`	CoDeSys-Browser-Kommandos	alle ecomatmobile-Controller PDM: CR10nn
`ifm_Errors_CRnnnn.INI`	CoDeSys-Fehler-Datei	alle ecomatmobile-Controller PDM: CR10nn

Target-Datei

2715

Dateiname	Funktion	verfügbar für:
`ifm_CRnnnn_Vxx.TRG`	Target-Datei	alle ecomatmobile-Controller BasicDisplay: CR0451 PDM: CR10nn

Steuerungskonfigurations-Datei

2716

Dateiname	Funktion	verfügbar für:
`ifm_CRnnnn_Vxxyyzz.CFG`	Steuerungskonfiguration	alle ecomatmobile-Controller BasicDisplay: CR0451 PDM: CR10nn

© ifm electronic gmbh

ifm-Gerätebibliotheken

2717

Dateiname	Funktion	verfügbar für:
ifm_CRnnnn_Vxxyyzz.LIB	gerätespezifische Bibliothek	alle ecomat*mobile*-Controller BasicDisplay: CR0451 PDM: CR10nn
ifm_CR0200_MSTR_Vxxyyzz.LIB	Bibliothek ohne Extended-Funktionen	ExtendedController: CR0200
ifm_CR0200_SMALL_Vxxyyzz.LIB	Bibliothek ohne Extended-Funktionen, reduzierter Funktionsumfang	ExtendedController: CR0200

ifm-CANopen-Hilfsbibliotheken Master/Slave

2718

Diese Bibliotheken setzen auf CoDeSys-Bibliotheken (3S-CANopen-Funktionen) auf und stellen sie dem Anwender übersichtlich zur Verfügung.

Dateiname	Funktion	verfügbar für:
ifm_CRnnnn_CANopenMaster_Vxxyyzz.LIB	CANopen Master Emergency- und Status-Handler	alle ecomat*mobile*-Controller *) PDM: CR10nn *)
ifm_CRnnnn_CANopenSlave_Vxxyyzz.LIB	CANopen Slave Emergency- und Status-Handler	alle ecomat*mobile*-Controller *) PDM: CR10nn *)
ifm_CANx_SDO_Vxxyyzz.LIB	CANopen SDO Read und SDO Write	PDM360: CR1050, CR1051 PDM360compact: CR1052, CR1053, CR1055, CR1056
ifm_CANopen_NT_Vxxyyzz.LIB	CANopen-Bausteine im CAN-Stack	BasicController: CR040n BasicDisplay: CR0451 PDM360NG: CR108n

*) jedoch NICHT für...
- BasicController: CR040n
- BasicDisplay: CR0451
- PDM360NG: CR108n

© ifm electronic gmbh

CoDeSys-CANopen-Bibliotheken

2719

Diese Bibliotheken sind für folgende Geräte NICHT verwendbar:
- BasicController: CR040n
- BasicDisplay: CR0451
- PDM360NG: CR108n

Dateiname	Funktion	verfügbar für:
3S_CanDrvOptTable.LIB ¹) 3S_CanDrvOptTableEx.LIB ²)	CANopen Treiber	alle ecomat*mobile*-Controller PDM360smart: CR1070, CR1071
3S_CanDrv.LIB ³)		PDM360: CR1050, CR1051 PDM360compact: CR1052, CR1053, CR1055, CR1056
3S_CANopenDeviceOptTable.LIB ¹) 3S_CANopenDeviceOptTableEx.LIB ²)	CANopen Slave-Treiber	alle ecomat*mobile*-Controller PDM360smart: CR1070, CR1071
3S_CANopenDevice.LIB ³)		PDM360: CR1050, CR1051 PDM360compact: CR1052, CR1053, CR1055, CR1056
3S_CANopenManagerOptTable.LIB ¹) 3S_CANopenManagerOptTableEx.LIB ²)	CANopen Netzwerkmanager	alle ecomat*mobile*-Controller PDM360smart: CR1070, CR1071
3S_CANopenManager.LIB ³)		PDM360: CR1050, CR1051 PDM360compact: CR1053, CR1056
3S_CANopenMasterOptTable.LIB ¹) 3S_CANopenMasterOptTableEx.LIB ²)	CANopen Master	alle ecomat*mobile*-Controller PDM360smart: CR1070, CR1071
3S_CANopenMaster.LIB ³)		PDM360: CR1050, CR1051 PDM360compact: CR1052, CR1053, CR1055, CR1056
3S_CANopenNetVarOptTable.LIB ¹) 3S_CANopenNetVarOptTableEx.LIB ²)	Treiber für Netzwerkvariablen	alle ecomat*mobile*-Controller PDM360smart: CR1070, CR1071
3S_CANopenNetVar.LIB ³)		PDM360: CR1050, CR1051 PDM360compact: CR1052, CR1053, CR1055, CR1056

¹) gültig für CRnn32 Target-Version bis V01, alle anderen Geräte bis V04
²) gültig für CRnn32 Target-Version ab V02, alle anderen Geräte ab V05
³) Für folgende Geräte gilt: diese Bibliothek ist funktionslos als Platzhalter enthalten:
- BasicController: CR040n
- BasicDisplay: CR0451
- PDM360NG: CR108n

© ifm electronic gmbh

spezielle ifm-Bibliotheken

2720

Dateiname	Funktion	verfügbar für:
ifm_RawCAN_NT_Vxxyyzz.LIB	CAN-Bausteine im CAN-Stack auf Basis Layer 2	BasicController: CR040n BasicDisplay: CR0451 PDM360NG: CR108n
ifm_J1939_NT_Vxxyyzz.LIB	J1939 Kommunikationsfunktionen im CAN-Stack	BasicController: CR040n BasicDisplay: CR0451 PDM360NG: CR108n
ifm_NetVarLib_NT_Vxxyyzz.LIB	zusätzlicher Treiber für Netzwerkvariablen	BasicController: CR040n BasicDisplay: CR0451 PDM360NG: CR108n
ifm_J1939_Vxxyyzz.LIB	J1939 Kommunikationsfunktionen	bis Target V04 für: CabinetController: CR0303 ClassicController: CR0020, CR0505 ExtendedController: CR0200 SafetyController: CR7020, CR7200, CR7505 SmartController: CR2500 PDM360smart: CR1070, CR1071
ifm_J1939_x_Vxxyyzz.LIB	J1939 Kommunikationsfunktionen	ab Target V05 für: CabinetController: CR0303 ClassicController: CR0020, CR0505 ExtendedController: CR0200 SafetyController: CR7nnn SmartController: CR2500 PDM360smart: CR1070, CR1071
ifm_CRnnnn_J1939_Vxxyyzz.LIB	J1939 Kommunikationsfunktionen	ClassicController: CR0032, CR0033 ExtendedController: CR0232, CR0233
ifm_PDM_J1939_Vxxyyzz.LIB	J1939 Kommunikationsfunktionen	PDM360: CR1050, CR1051 PDM360compact: CR1052, CR1053, CR1055, CR1056
ifm_CANx_LAYER2_Vxxyyzz.LIB	CAN-Bausteine auf Basis Layer 2: CAN Transmit, CAN Receive	PDM360: CR1050, CR1051 PDM360compact: CR1052, CR1053, CR1055, CR1056
ifm_CAN1E_Vxxyyzz.LIB	Stellt den CAN-Bus von 11 Bit auf 29 Bit um	bis Target V04 für: PDM360smart: CR1070, CR1071

© ifm electronic gmbh

Dateiname	Funktion	verfügbar für:
ifm_CAN1_EXT_Vxxyyzz.LIB	Stellt den CAN-Bus von 11 Bit auf 29 Bit um	ab Target V05 für: CabinetController: CR030n ClassicController: CR0020, CR0505 ExtendedController: CR0200 Platinensteuerung: CS0015 SafetyController: CR7020, CR7021, CR7200, CR7201, CR7505, CR7506 SmartController: CR25nn PDM360smart: CR1070, CR1071
ifm_CAMERA_O2M_Vxxyyzz.LIB	Kamera-Funktionen	PDM360: CR1051
CR2013AnalogConverter.LIB	Analogwertkonvertierung für E/A-Modul CR2013	alle ecomat*mobile*-Controller PDM: CR10nn
ifm_Hydraulic_16bitOS04_Vxxyyzz.LIB	Hydraulikfunktionen für Controller	bis Target V04 für: ClassicController: CR0020, CR0505 ExtendedController: CR0200 SafetyController: CR7020, CR7200, CR7505 SmartController: CR25nn
ifm_Hydraulic_16bitOS05_Vxxyyzz.LIB	Hydraulikfunktionen für Controller	ab Target V05 für: ClassicController: CR0020, CR0505 ExtendedController: CR0200 SafetyController: CR7020, CR7021, CR7200, CR7201, CR7505, CR7506 SmartController: CR25nn
ifm_Hydraulic_32bit_Vxxyyzz.LIB	Hydraulikfunktionen für Controller	ClassicController: CR0032, CR0033 ExtendedController: CR0232, CR0233
ifm_Hydraulic_CR0303_Vxxyyzz.LIB	Hydraulikfunktionen für Controller	ab Target V05 für: CabinetController: CR0303
ifm_SafetyIO_Vxxyyzz.LIB	Sicherheitsfunktionen	SafetyController: CR7nnn
ifm_PDM_UTIL_Vxxyyzz.LIB	Hilfsfunktionen PDM	PDM360: CR1050, CR1051 PDM360compact: CR1052, CR1053, CR1055, CR1056
ifm_PDMng_UTIL_Vxxyyzz.LIB	Hilfsfunktionen PDM	PDM360NG: CR108n
ifm_PDMsmart_UTIL_Vxxyyzz.LIB	Hilfsfunktionen PDM	PDM360smart: CR1070, CR1071
ifm_PDM_Input_Vxxyyzz.LIB	alternative Eingabefunktionen PDM	PDM: CR10nn
ifm_CR107n_Init_Vxxyyzz.LIB	Initialisierungsfunktion PDM360smart	PDM360smart: CR1070, CR1071
ifm_PDM_File_Vxxyyzz.LIB	Dateifunktionen PDM360	PDM360: CR1050, CR1051 PDM360compact: CR1052, CR1053, CR1055, CR1056 PDM360NG: CR108n
ifm_PDM360NG_linux_syscall_asynch.LIB	Linux-Kommandos an das System senden	PDM360NG: CR108n
ifm_PDM360NG_USB_Vxxyyzz.LIB	Geräte an der USB-Schnittstelle verwalten	PDM360NG: CR108n
ifm_PDM360NG_USB_LL_Vxxyyzz.LIB	Hilfsbibliothek für ifm_PDM360NG_USB_Vxxyyzz.LIB	PDM360NG: CR108n

© ifm electronic gmbh

Dateiname	Funktion	verfügbar für:
`Instrumente_x.LIB`	vordefinierte Anzeige-Instrumente	PDM: CR10nn
`Symbols_x.LIB`	vordefinierte Symbole	PDM360: CR1050, CR1051 PDM360compact: CR1052, CR1053, CR1055, CR1056
`Segment_x.LIB`	vordefinierte 7-Segment-Anzeigen	PDM360: CR1050, CR1051 PDM360compact: CR1052, CR1053, CR1055, CR1056

Weitere Bibliotheken auf Anfrage.

© ifm electronic gmbh

© ifm electronic gmbh

13 Begriffe und Abkürzungen

A

Adresse

Das ist der „Name“ des Teilnehmers im Bus. Alle Teilnehmer benötigen eine unverwechselbare, eindeutige Adresse, damit der Austausch der Signale fehlerfrei funktioniert.

Anforderungsrate rd

Die Anforderungsrate r_d ist die Häufigkeit je Zeiteinheit von Anforderungen an eine sicherheitsgerichtete Reaktion eines SRP/CS.

Anleitung

Übergeordnetes Wort für einen der folgenden Begriffe:
Montageanleitung, Datenblatt, Benutzerinformation, Bedienungsanleitung, Gerätehandbuch, Installationsanleitung, Onlinehilfe, Systemhandbuch, Programmierhandbuch, usw.

Applikations-Software

Software, die speziell für die Applikation (Anwendung) vom Hersteller in die Maschine programmiert wird. Die Software enthält üblicherweise logische Sequenzen, Grenzwerte und Ausdrücke zum Steuern der entsprechenden Ein- und Ausgänge, Berechnungen und Entscheidungen.

Für sicherheitsrelevante Teile von Steuerungen (→SRP/CS) müssen spezielle Anforderungen erfüllt sein.
→ Programmiersprache, sicherheitsrelevant

Architektur

Spezifische Konfiguration von Hardware- und Software-Elementen in einem System.

Ausfall

Ausfall ist die Beendigung der Fähigkeit einer Einheit, eine geforderte Funktion zu erfüllen.

Nach einem Ausfall hat die Einheit einen Fehler. Der Ausfall ist ein Ereignis, der Fehler ein Zustand.

Der so definierte Begriff kann nicht auf Einheiten angewendet werden, die nur aus Software bestehen.

Ausfall, gefahrbringend

Ein gefahrbringender Ausfall hat das Potential, das SRP/CS in einen gefährlichen Zustand oder eine Fehlfunktion zu bringen. Ob dieses Potential bemerkt werden kann oder nicht, hängt von der Architektur des Systems ab. In einem redundanten System wird ein gefährlicher Hardware-Ausfall weniger wahrscheinlich zu einem gefährlichen Ausfall des Gesamtsystems führen.

Ausfall, systematischer

Ein systematischer Ausfall ist ein Ausfall mit deterministischem (nicht zufälligem) Bezug zu einer bestimmten Ursache. Der systematische Ausfall kann nur beseitigt werden durch Änderung des Entwurfs oder des Herstellprozesses, Betriebsverfahren, Dokumentation oder zugehörigen Faktoren.

Eine Instandsetzung ohne Änderung des Systems wird den Grund des systematischen Ausfalls in der Regel nicht beseitigen.

B

Baud

Baud, Abk.: Bd = Maßeinheit für die Geschwindigkeit bei der Datenübertragung. Baud ist nicht zu verwechseln mit "bits per second" (bps, Bit/s). Baud gibt zwar die Anzahl von Zustandsänderungen (Schritte, Takte) pro Sekunde auf einer Übertragungsstrecke an. Aber es ist nicht festgelegt, wie viele Bits pro Schritt übertragen werden. Der Name Baud geht auf den französischen Erfinder J. M. Baudot zurück, dessen Code für Telexgeräte verwendet wurde.

1 MBd = 1024 x 1024 Bd = 1 048 576 Bd

Bestimmungsgemäße Verwendung

Das ist die Verwendung eines Produkts in Übereinstimmung mit den in der Anleitung bereitgestellten Informationen.

© ifm electronic gmbh

Betriebsdauer, mittlere

Mean **t**ime **b**etween **f**ailures (MTBF) = mittlere Betriebsdauer zwischen Ausfällen.
Ist der Erwartungswert der Betriebsdauer zwischen zwei aufeinanderfolgenden Ausfällen von Einheiten, die instand gesetzt werden.

Für Einheiten, die NICHT instandgesetzt werden, ist der Erwartungswert (Mittelwert) der Verteilung von Lebensdauern die mittlere Lebensdauer →MTTF.

Betriebssystem

Grundprogramm im Gerät, stellt die Verbindung her zwischen der Hardware des Gerätes und der Anwender-Software.

Bootloader

Im Auslieferungszustand enthalten ecomat*mobile*-Geräte nur den Bootloader. Der Bootloader ist ein Startprogramm, mit dem das Betriebssystem (= Laufzeitsystem) und das Applikations-Programm auf dem Gerät nachgeladen werden können.

Der Bootloader enthält Grundroutinen...
- zur Kommunikation der Hardware-Module untereinander,
- zum Nachladen des Betriebssystems.

Der Bootloader ist das erste Software-Modul, das im Gerät gespeichert sein muss.

Bus

Serielle Datenübertragung mehrerer Teilnehmer an derselben Leitung.

C

CAN

CAN = **C**ontroller **A**rea **N**etwork

CAN gilt als Feldbussystem für größere Datenmengen, das prioritätengesteuert arbeitet. Gibt es in verschiedenen Varianten z.B. als "CANopen" oder "CAN in Automation (CiA)."

CAN-Stack

CAN-Stack = Stapel von CAN-Datenübertragungs-Aufträgen.

CCF

Common **c**ause **f**ailure = Ausfall in Folge von gemeinsamer Ursache
Ausfälle verschiedener Einheiten aufgrund eines gemeinsamen Ereignisses, wobei diese Ausfälle nicht auf gegenseitige Ursachen beruhen.

CiA

CiA = CAN in Automation e.V.

Anwender- und Herstellerorganisation in Deutschland / Erlangen. Definitions- und Kontrollorgan für CAN und CAN-basierende Netzwerkprotokolle.

Homepage → http://www.can-cia.org

CiA DS 304

DS = **D**raft **S**tandard

CAN-Geräteprofil CANopen-Safety für sicherheitsgerichtete Kommunikation.

CiA DS 401

DS = **D**raft **S**tandard

CAN-Geräteprofil für digitale und analoge E/A-Baugruppen

CiA DS 402

DS = **D**raft **S**tandard

CAN-Geräteprofil für Antriebe

CiA DS 403

DS = **D**raft **S**tandard

CAN-Geräteprofil für Bediengeräte

CiA DS 404

DS = **D**raft **S**tandard

CAN-Geräteprofil für Messtechnik und Regler

CiA DS 405

DS = **D**raft **S**tandard

Spezifikation zur Schnittstelle zu programmierbaren Steuerungen (IEC 61131-3)

© ifm electronic gmbh

CiA DS 406

DS = **D**raft **S**tandard

CAN-Geräteprofil für Drehgeber / Encoder

CiA DS 407

DS = **D**raft **S**tandard

CAN-Applikations-Profil für den öffentlichen Nahverkehr

COB-ID

COB = **C**ommunication **Ob**ject = Kommunikationsobjekt
ID = **Id**entifier = Kennung

Über den COB-ID unterscheiden die Teilnehmer die verschiedenen auszutauschenden Nachrichten.

CoDeSys

CoDeSys ist eingetragene Marke der 3S – Smart Software Solutions GmbH, Deutschland

"CoDeSys for Automation Alliance" vereinigt Firmen der Automatisierungsindustrie, deren Hardwaregeräte alle mit dem weit verbreiteten IEC 61131-3 Entwicklungswerkzeug CoDeSys programmiert werden.

Homepage → http://www.3s-software.com

CRC

CRC = **C**yclic **R**edundancy **C**heck = zyklische Redundanzprüfung

CRC ist ein Verfahren aus der Informationstechnik zur Bestimmung eines Prüfwerts für Daten, um Fehler bei der Übertragung oder Duplizierung von Daten erkennen zu können.

Vor Beginn der Übertragung eines Blocks der Daten wird ein CRC-Wert berechnet. Nach Abschluss der Transaktion wird am Zielort der CRC-Wert erneut berechnet. Anschließend werden diese beiden Prüfwerte verglichen.

D

DC

Direct **C**urrent = Gleichstrom

DC

Diagnostic **C**overage = Diagnose-Deckungsgrad
Der Diagnose-Deckungsgrad ist das Maß für die Wirksamkeit der Diagnose als Verhältnis der Ausfallrate der bemerkten gefahrbringenden Ausfälle und der Ausfallrate der gesamten gefahrbringenden Ausfälle:

Formel: DC = Ausfallrate bemerkte gefahrbringende Ausfälle / Ausfallrate gesamte gefahrbringende Ausfälle

Bezeichnung	Bereich
kein	DC < 60 %
niedrig	60 % ≤ DC < 90 %
mittel	90 % ≤ DC < 99 %
hoch	99 % ≤ DC

Tabelle: Diagnose-Deckungsgrad DC

Für die in der Tabelle gezeigten Grenzwerte wird eine Genauigkeit von 5 % angenommen.

Der Diagnose-Deckungsgrad kann für das gesamte sicherheitsgerichtete System ermittelt werden oder nur für Teile des sicherheitsgerichteten Systems.

Diagnose

Bei der Diagnose wird der "Gesundheitszustand" des Gerätes geprüft. Es soll festgestellt werden, ob und gegebenenfalls welche Fehler im Gerät vorhanden sind.

Je nach Gerät können auch die Ein- und Ausgänge auf einwandfreie Funktion überwacht werden:
- Drahtbruch,
- Kurzschluss,
- Wert außerhalb des Sollbereichs.

Zur Diagnose können Konfigurations-Dateien herangezogen werden, die während des "normalen" Betriebs des Gerätes erzeugt wurden.

Der korrekte Start der Systemkomponenten wird während der Initialisierungs- und Startphase überwacht.

Zur weiteren Diagnose können auch Selbsttests durchgeführt werden.

© ifm electronic gmbh

Diagnose-Deckungsgrad

Diagnostic **C**overage = Diagnose-Deckungsgrad
Der Diagnose-Deckungsgrad ist das Maß für die Wirksamkeit der Diagnose als Verhältnis der Ausfallrate der bemerkten gefahrbringenden Ausfälle und der Ausfallrate der gesamten gefahrbringenden Ausfälle:

Formel: DC = Ausfallrate bemerkte gefahrbringende Ausfälle / Ausfallrate gesamte gefahrbringende Ausfälle

Bezeichnung	Bereich
kein	DC < 60 %
niedrig	60 % ≤ DC < 90 %
mittel	90 % ≤ DC < 99 %
hoch	99 % ≤ DC

Tabelle: Diagnose-Deckungsgrad DC

Für die in der Tabelle gezeigten Grenzwerte wird eine Genauigkeit von 5 % angenommen.

Der Diagnose-Deckungsgrad kann für das gesamte sicherheitsgerichtete System ermittelt werden oder nur für Teile des sicherheitsgerichteten Systems.

Dither

to dither (engl.) = schwanken / zittern

Dither ist ein Bestandteil der PWM-Signale zum Ansteuern von Hydraulik-Ventilen. Für die elektromagnetischen Antriebe von Hydraulik-Ventilen hat sich herausgestellt, dass sich die Ventile viel besser regeln lassen, wenn das Steuersignal (PWM-Impulse) mit einer bestimmten Frequenz der PWM-Frequenz überlagert wird. Diese Dither-Frequenz muss ein ganzzahliger Teil der PWM-Frequenz sein.
→ Kapitel *Was ist der Dither?*

diversitär

Unter Diversität (Vielfalt) versteht man in der Technik eine Strategie zur Erhöhung der Ausfallsicherheit.

Dabei werden Systeme redundant ausgelegt, allerdings werden bewusst verschiedene Realisierungen und keine baugleichen Einzelsysteme verwendet. Man geht davon aus, dass Systeme, die das Gleiche leisten, aber unterschiedlich realisiert sind, auch gegen unterschiedliche Störungen empfindlich oder unempfindlich sind und daher möglichst nicht alle gleichzeitig ausfallen.

Die konkrete Realisierung kann je nach Einsatzgebiet und geforderter Sicherheit unterschiedlich aussehen:

- Verwendung von Bauteilen verschiedener Hersteller,
- Nutzung unterschiedlicher Protokolle zur Steuerung von Geräten,
- Verwendung komplett unterschiedlicher Technologien, beispielsweise einer elektrischen und einer pneumatischen Steuerung,
- Verwendung unterschiedlicher Messmethoden (Strom, Spannung),
- zwei Kanäle mit gegenläufigen Werteverläufen:
  Kanal A: 0...100 %
  Kanal B: 100...0 %

DRAM

DRAM = **D**ynamic **R**andom **A**ccess **M**emory

Technologie für einen elektronischen Speicherbaustein mit wahlfreiem Zugriff (Random Access Memory, RAM). Das speichernde Element ist dabei ein Kondensator, der entweder geladen oder entladen ist. Über einen Schalttransistor wird er zugänglich und entweder ausgelesen oder mit neuem Inhalt beschrieben. Der Speicherinhalt ist flüchtig: die gespeicherte Information geht bei fehlender Betriebsspannung oder zu später Wiederauffrischung verloren.

DTC

DTC = **D**iagnostic **T**rouble **C**ode = Fehler-Code
Störungen und Fehler werden über zugeordnete Nummern – den DTCs – verwaltet und gemeldet.

E

ECU

(1) **E**lectronic **C**ontrol **U**nit = Steuergerät oder Mikrocontroller
(2) **E**ngine **C**ontrol **U**nit = Steuergerät eines Motors

© ifm electronic gmbh

EDS-Datei

EDS = **E**lectronic **D**ata **S**heet = elektronisch hinterlegtes Datenblatt, z.B. für:

- Datei für das Objektverzeichnis im Master
- CANopen-Gerätebeschreibungen

Via EDS können vereinfacht Geräte und Programme ihre Spezifikationen austauschen und gegenseitig berücksichtigen.

Embedded Software

System-Software, Grundprogramm im Gerät, praktisch das Betriebssystem.

Die Firmware stellt die Verbindung her zwischen der Hardware des Gerätes und der Anwender-Software. Diese Software wird vom Hersteller der Steuerung als Teil des Systems geliefert und kann vom Anwender nicht verändert werden.

EMCY

Abkürzung für Emergency (engl.) = Notfall

EMV

EMV = **E**lektro-**M**agnetische **V**erträglichkeit

Gemäß der EG-Richtlinie (2004/108/EG) zur elektromagnetischen Verträglichkeit (kurz EMV-Richtlinie) werden Anforderungen an die Fähigkeit von elektrischen und elektronischen Apparaten, Anlagen, Systemen oder Bauteilen gestellt, in der vorhandenen elektromagnetischen Umwelt zufriedenstellend zu arbeiten. Die Geräte dürfen ihre Umgebung nicht stören und dürfen sich von äußerlichen elektromagnetischen Störungen nicht ungünstig beeinflussen lassen.

Erstfehler-Eintrittszeit

Das ist die Zeit bis zum ersten Versagen eines Sicherheitselements.

Im Zeitraum von maximal 30 s überprüft das Betriebssystem mittels interner Überwachungs- und Testroutinen die Steuerung.

Diese „Testzykluszeit“ muss kleiner sein als die statistische Erstfehler-Eintrittszeit für die Applikation.

Ethernet

Das Ethernet ist eine weit verbreitete, herstellerneutrale Technologie, mit der im Netzwerk Daten mit einer Geschwindigkeit von 10 oder 100 Millionen Bit pro Sekunde (Mbps) übertragen werden können. Das Ethernet gehört zu der Familie der sogenannten „bestmöglichen Datenübermittlung“ auf einem nicht exklusiven Übertragungsmedium. 1972 entwickelt, wurde das Konzept 1985 als IEEE 802.3 spezifiziert.

EUC

EUC = „equipment under control" (kontrollierte Einrichtung)

EUC ist eine Einrichtung, Maschine, Gerät oder Anlage, verwendet zur Fertigung, Stoffumformung, zum Transport, zu medizinischen oder anderen Tätigkeiten (→ IEC 61508-4, Abschnitt 3.2.3). Das EUC umfasst also alle Einrichtungen, Maschinen, Geräte oder Anlagen, die Gefährdungen verursachen können und für die sicherheitsgerichtete Systeme erforderlich sind.

Falls eine vernünftigerweise vorhersehbare Aktivität oder Inaktivität zu durch das EUC verursachten Gefährdungen mit unvertretbarem Risiko führt, sind Sicherheitsfunktionen erforderlich, um einen sicheren Zustand für das EUC zu erreichen oder aufrecht zu erhalten. Diese Sicherheitsfunktionen werden durch ein oder mehrere sicherheitsgerichtete Systeme ausgeführt.

F

Fehlanwendung

Das ist die Verwendung eines Produkts in einer Weise, die vom Konstrukteur nicht vorgesehen ist. Eine Fehlanwendung führt meist zu einer Gefährdung von Personen oder Sachen.

Vor vernünftigerweise, vorhersehbaren Fehlanwendungen muss der Hersteller des Produkts in seinen Benutzerinformationen warnen.

© ifm electronic gmbh

Fehler

Ein Fehler ist die Unfähigkeit einer Einheit, eine geforderte Funktion auszuführen.

Kein Fehler ist diese Unfähigkeit während vorbeugender Wartung oder anderer geplanter Handlungen oder aufgrund des Fehlers externer Mittel.

Ein Fehler ist oft das Resultat eines Ausfalls der Einheit selbst, kann aber ohne vorherigen Ausfall bestehen.

In der ISO 13849-1 ist mit "Fehler" der "zufällige Fehler" gemeint.

Fehler-Toleranzzeit

Das ist die maximale Zeit, die zwischen dem Entstehen eines Fehlers und der Einnahme des sicheren Zustandes in der Applikation vergehen darf, ohne dass eine Gefahr für Personen zu befürchten ist.

Dabei ist die maximale Zykluszeit des Applikations-Programms (im ungünstigsten Fall 100 ms, → *Verhalten des Watchdog*, → Seite 70) und die möglichen Verzögerungs- und Reaktionszeiten durch Abschaltglieder zu berücksichtigen.

Die sich daraus ergebende Gesamtzeit muss kleiner sein als die Fehler-Toleranzzeit der Applikation.

FiFo

FiFo (**F**irst **I**n, **F**irst **O**ut) = Arbeitsweise des Stapelspeichers: Das Datenpaket, das zuerst in den Stapelspeicher geschrieben wurde, wird auch als erstes gelesen. Pro Identifier steht ein solcher Zwischenspeicher (als Warteschlange) zur Verfügung.

Firmware

System-Software, Grundprogramm im Gerät, praktisch das Betriebssystem.

Die Firmware stellt die Verbindung her zwischen der Hardware des Gerätes und der Anwender-Software. Diese Software wird vom Hersteller der Steuerung als Teil des Systems geliefert und kann vom Anwender nicht verändert werden.

Flash-Speicher

Flash-ROM (oder Flash-EPROM oder Flash-Memory) kombiniert die Vorteile von Halbleiterspeicher und Festplatten. Wie jeder andere Halbleiterspeicher kommt Flash-Speicher ohne bewegliche Teile aus. Und die Daten bleiben wie bei einer Festplatte auch nach dem Ausschalten erhalten.

Der Flash-ROM hat sich aus dem EEPROM (**E**lectrical **E**rasable and **P**rogrammable **R**ead-**O**nly **M**emory) entwickelt. Beim Flash-ROM ist die Speicherung von Daten funktionell identisch wie beim EEPROM. Die Daten werden allerdings wie bei einer Festplatte blockweise in Datenblöcken zu 64, 128, 256, 1024, ... Byte zugleich geschrieben und gelöscht.

Vorteile von Flash-Speicher

- Die gespeicherten Daten bleiben auch bei fehlender Versorgungsspannung erhalten.
- Wegen fehlender beweglicher Teile ist Flash geräuschlos, unempfindlich gegen Erschütterungen und magnetische Felder.
- Im Vergleich zu Festplatten haben Flash-Speicher eine sehr kurze Zugriffszeit. Lese- und Schreibgeschwindigkeit sind über den gesamten Speicherbereich weitestgehend konstant.
- Die erreichbare Speichergröße ist durch die einfache und platzsparende Anordnung der Speicherzellen nach oben offen.

Nachteile von Flash-Speicher

- Begrenzte Zahl von Schreib- bzw. Löschvorgängen, die eine Speicherzelle vertragen kann:
  - Multi-Level-Cells: typ. 10 000 Zyklen
  - Single-Level-Cells: typ. 100 000 Zyklen
- Da ein Schreibvorgang Speicherblöcke zwischen 16 und 128 kByte gleichzeitig beschreibt, werden auch Speicherzellen beansprucht, die gar keiner Veränderung bedürfen.

FMEA

FMEA = **F**ailure **M**ode and **E**ffects **A**nalysis = **F**ehler-**M**öglichkeits- und **E**influss-**A**nalyse

Methode der Zuverlässigkeitstechnik, um potenzielle Schwachstellen zu finden. Im Rahmen des Qualitäts- oder Sicherheitsmanagements wird die FMEA zur

© ifm electronic gmbh

Fehlervermeidung und Erhöhung der technischen Zuverlässigkeit vorbeugend eingesetzt.

FRAM

FRAM, oder auch FeRAM, bedeutet **Fe**rroelectric **R**andom **A**ccess **M**emory. Der Speicher- und Löschvorgang erfolgt durch eine Polarisationsänderung in einer ferroelektrischen Schicht.

Vorteile von FRAM gegenüber herkömmlichen Festwertspeichern:

- nicht flüchtig,
- kompatibel zu gängigen EEPROMs, jedoch:
- Zugriffszeit ca. 100 ns,
- fast unbegrenzt viele Zugriffszyklen möglich.

Funktionale Sicherheit

Teil der Gesamtsicherheit, bezogen auf das →EUC und das EUC-Leit- oder Steuerungssystem, die von der korrekten Funktion des elektrischen oder elektronischen sicherheitsgerichteten Systems, sicherheitsgerichteten Systemen anderer Technologien und externer Einrichtungen zur Risikominderung abhängt.

G

Gebrauchsdauer Tm

Die Gebrauchsdauer T_M ist der Zeitraum, der die vorgegebene Verwendung der SRP/CS abdeckt.

Gefährdung

Mit Gefährdung bezeichnet man eine potentielle Schadensquelle.

Man unterscheidet den Ursprung der Gefährdung, z.B.:
- mechanische Gefährdung,
- elektrische Gefährdung,

oder die Art des zu erwartenden Schadens, z.B.:
- Gefährdung durch elektrischen Schlag,
- Gefährdung durch Schneiden,
- Gefährdung durch Vergiftung.

Die Gefährdung im Sinne dieser Definition ist bei der bestimmungsgemäßen Verwendung der Maschine entweder dauerhaft vorhanden, z.B.:
- Bewegung von gefährdenden beweglichen Teilen,
- Lichtbogen beim Schweißen,
- ungesunde Körperhaltung,
- Geräusch-Emission,
- hohe Temperatur,

oder die Gefährdung kann unerwartet auftreten, z.B.:
- Explosion,
- Gefährdung durch Quetschen als Folge eines unbeabsichtigten / unerwarteten Anlaufs,
- Herausschleudern als Folge eines Bruchs,
- Stürzen als Folge von Geschwindigkeitsänderung.

H

Heartbeat

Heartbeat (engl.) = Herzschlag

Die Teilnehmer senden regelmäßig kurze Signale. So können die anderen Teilnehmer prüfen, ob ein Teilnehmer ausgefallen ist. Dazu ist kein Master erforderlich.

HMI

HMI = **H**uman **M**achine **I**nterface = Mensch-Maschine-Schnittstelle

I

ID - Identifier

ID = **Id**entifier = Kennung

Name zur Unterscheidung der an einem System angeschlossenen Geräte / Teilnehmer oder der zwischen den Teilnehmern ausgetauschten Nachrichtenpakete.

IEC-User-Zyklus

IEC-User-Zyklus = SPS-Zyklus im CoDeSys-Applikations-Programm.

IP-Adresse

IP = **I**nternet **P**rotocol = Internet-Protokoll

Die IP-Adresse ist eine Nummer, die zur eindeutigen Identifizierung eines Internet-Teilnehmers notwendig ist. Zur besseren

© ifm electronic gmbh

Übersicht wird die Nummer in 4 dezimalen Werten geschrieben, z. B. 127.215.205.156.

ISO 11898

Norm: "Straßenfahrzeuge – CAN-Protokoll"

Teil 1: "Bit-Übertragungsschicht und physikalische Zeichenabgabe"

Teil 2: "High-speed medium access unit"

Teil 3: "Fehlertolerante Schnittstelle für niedrige Geschwindigkeiten"

Teil 4: "Zeitgesteuerte Kommunikation"

Teil 5: "High-speed medium access unit with low-power mode"

ISO 11992

Norm: "Straßenfahrzeuge – Austausch von digitalen Informationen über elektrische Verbindungen zwischen Zugfahrzeugen und Anhängefahrzeugen"

Teil 1: "Bit-Übertragungsschicht und Sicherungsschicht"

Teil 2: "Anwendungsschicht für die Bremsausrüstung"

Teil 3: "Anwendungsschicht für andere als die Bremsausrüstung"

Teil 4: "Diagnose"

ISO 16845

Norm: "Straßenfahrzeuge – Steuergerätenetz (CAN) – Prüfplan zu Konformität"

K

Kategorie (CAT)

Einstufung der sicherheitsrelevante Teile einer Steuerung bezüglich ihres Widerstandes gegen Fehler und ihres nachfolgenden Verhaltens bei einem Fehler. Diese Sicherheit wird erreicht durch die Struktur der Anordnung der Teile, die Fehlererkennung und/oder ihre Zuverlässigkeit. (→ EN 954)

Klemme 15

Klemme 15 ist in Fahrzeugen die vom Zündschloss geschaltete Plusleitung.

L

Lebensdauer, mittlere

Mean **t**ime **t**o **d**angerous **f**ailure = erwartete mittlere Dauer bis zum gefahrbringenden Ausfall.

Bezeichnung	Bereich
niedrig	3 Jahre $\leq$ $MTTF_d$ < 10 Jahre
mittel	10 Jahre $\leq$ $MTTF_d$ < 30 Jahre
hoch	30 Jahre $\leq$ $MTTF_d$ $\leq$ 100 Jahre

Tabelle: Mittlere Zeit jedes Kanals bis zum gefahrbringenden Ausfall $MTTF_d$

LED

LED = **L**ight **E**mitting **D**iode = Licht aussendende Diode

Leuchtdiode, auch Luminiszenzdiode, ein elektronisches Element mit hoher, farbiger Leuchtkraft auf kleinem Volumen bei vernachlässigbarer Verlustleistung.

Link

Ein Link ist ein Querverweis zu einer anderen Stelle im Dokument oder auf ein externes Dokument.

LSB

Least **S**ignificant **B**it/Byte = Niederwertigstes Bit/Byte in einer Reihe von Bit/Bytes.

M

MAC-ID

MAC = **M**anufacturer‘s **A**ddress **C**ode = Hersteller-Seriennummer

→ID = **Id**entifier = Kennung

Jede Netzwerkkarte verfügt über eine so genannte MAC-Adresse, ein unverwechselbarer, auf der ganzen Welt einzigartiger Zahlencode – quasi eine Art Seriennummer. So eine MAC-Adresse ist eine Aneinanderreihung von 6 Hexadezimalzahlen, etwa "00-0C-6E-D0-02-3F".

© ifm electronic gmbh

Master

Wickelt die komplette Organisation auf dem Bus ab. Der Master entscheidet über den zeitlichen Buszugriff und fragt die →Slaves zyklisch ab.

MMI

MMI = **M**ensch-**M**aschine-**I**nterface
→ *HMI*, → Seite 339

MRAM

MRAM bedeutet **M**agnetoresistive **R**andom **A**ccess **M**emory. Die Informationen werden mit magnetischen Ladungselementen gespeichert. Dabei wird die Eigenschaft bestimmter Materialien ausgenutzt, die ihren elektrischen Widerstand unter dem Einfluss magnetischer Felder ändern.

Vorteile von MRAM gegenüber herkömmlichen Festwertspeichern:

- nicht flüchtig (wie FRAM), jedoch:
- Zugriffszeit nur ca. 35 ns,
- unbegrenzt viele Zugriffszyklen möglich.

MSB

Most **S**ignificant **B**it/Byte = Höchstwertiges Bit/Byte einer Reihe von Bits/Bytes.

MTBF

Mean **t**ime **b**etween **f**ailures (MTBF) = mittlere Betriebsdauer zwischen Ausfällen.
Ist der Erwartungswert der Betriebsdauer zwischen zwei aufeinanderfolgenden Ausfällen von Einheiten, die instand gesetzt werden.

Für Einheiten, die NICHT instandgesetzt werden, ist der Erwartungswert (Mittelwert) der Verteilung von Lebensdauern die mittlere Lebensdauer →MTTF.

MTTF

Mean **t**ime **t**o **f**ailure (MTTF) = mittlere Dauer bis zum Ausfall oder: mittlere Lebensdauer.

MTTFd

Mean **t**ime **t**o **d**angerous **f**ailure = erwartete mittlere Dauer bis zum gefahrbringenden Ausfall.

Bezeichnung	Bereich
niedrig	3 Jahre ≤ $MTTF_d$ < 10 Jahre
mittel	10 Jahre ≤ $MTTF_d$ < 30 Jahre
hoch	30 Jahre ≤ $MTTF_d$ ≤ 100 Jahre

Tabelle: Mittlere Zeit jedes Kanals bis zum gefahrbringenden Ausfall $MTTF_d$

Muting

Mit Muting bezeichnet man die vorübergehende und automatische Unterdrückung einer Sicherheitsfunktion durch das SRP/CS.

Beispiel: Der Sicherheits-Lichtvorhang ist überbrückt, wenn die schließenden Werkzeuge unter einen fingersicheren Abstand zueinander gelangt sind. Die bedienende Person kann nun gefahrlos an die Maschine herantreten und das Werkstück führen.

N

NMT

NMT = **N**etwork **M**anagemen**t** = Netzwerk-Verwaltung (hier: im CAN-Bus)

Der NMT-Master steuert die Betriebszustände der NMT-Slaves.

Node

Node (engl.) = Knoten. Damit ist ein Teilnehmer im Netzwerk gemeint.

Node Guarding

Node (engl.) = Knoten, hier: Netzwerkteilnehmer
Guarding (engl.) = Schutz

Parametrierbare, zyklische Überwachung von jedem entsprechend konfigurierten Slave. Der Master prüft, ob die Slaves rechtzeitig antworten. Die Slaves prüfen, ob der Master regelmäßig anfragt. Somit können ausgefallene Netzwerkteilnehmer schnell erkannt und gemeldet werden.

© ifm electronic gmbh

O

Obj / Objekt

Oberbegriff für austauschbare Daten / Botschaften innerhalb des CANopen-Netzwerks.

Objektverzeichnis

Das **Ob**jekt**v**erzeichnis OBV enthält alle CANopen-Kommunikationsparameter eines Gerätes, sowie gerätespezifische Parameter und Daten.

OBV

Das **Ob**jekt**v**erzeichnis OBV enthält alle CANopen-Kommunikationsparameter eines Gerätes, sowie gerätespezifische Parameter und Daten.

operational

Operational (engl.) = betriebsbereit

Betriebszustand eines CANopen-Teilnehmers. In diesem Modus können SDOs, NMT-Kommandos und PDOs übertragen werden.

P

PC-Karte

→ PCMCIA-Karte

PCMCIA-Karte

PCMCIA = Personal Computer Memory Card International Association, ein Standard für Erweiterungskarten mobiler Computer.
Seit der Einführung des Cardbus-Standards 1995 werden PCMCIA-Karten auch als PC-Karte (engl.: PC Card) bezeichnet.

PDM

PDM = **P**rocess and **D**ialog **M**odule = **P**rozess- und **D**ialog-**M**onitor

Gerät zur Kommunikation des Bedieners mit der Maschine / Anlage.

PDO

PDO = **P**rocess **D**ata **O**bject = Nachrichten-Objekt mit Prozessdaten.

Die zeitkritischen Prozessdaten werden mit Hilfe der "Process Data Objects" (PDOs) übertragen. Die PDOs können beliebig zwischen den einzelnen Knoten ausgetauscht werden (PDO-Linking). Zusätzlich wird festgelegt, ob der Datenaustausch ereignisgesteuert (asynchron) oder synchronisiert erfolgen soll. Je nach der Art der zu übertragenden Daten kann die richtige Wahl der Übertragungsart zu einer erheblichen Entlastung des CAN-Bus führen.

Diese Dienste sind vom Protokoll her nicht bestätigte Dienste, d.h. es gibt keine Kontrolle, ob die Nachricht auch beim Empfänger ankommt. Netzwerkvariablen-Austausch entspricht einer "1-zu-n-Verbindung" (1 Sender zu n Empfängern).

PDU

PDU = **P**rotocol **D**ata **U**nit = Protokoll-Daten-Einheit

Die PDU ist ein Begriff aus dem CAN-Protokoll SAE J1939. Sie bezeichnet einen Bestandteil der Ziel- oder Quelladresse.

Performance-Level

Performance-**L**evel
Ist nach ISO 13849-1 eine Einstufung (PL a...e) der Fähigkeit von sicherheitsrelevanten Teilen einer Steuerung, eine Sicherheitsfunktion unter vorhersehbaren Bedingungen auszuführen.
→ Kapitel *Performance-Level PL*

PES

Programable **e**lectronic **s**ystem = Programmierbares elektronisches System
Ein programmierbares elektronisches System ist ein System ...
- zur Steuerung, zum Schutz oder zur Überwachung,
- auf der Basis einer oder mehrerer programmierbarer Geräte,
- einschließlich aller Elemente dieses Systems, wie Ein- und Ausgabegeräte.

© ifm electronic gmbh

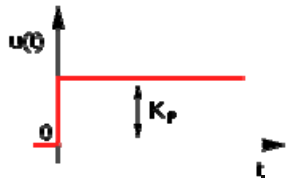
PGN

PGN = **P**arameter **G**roup **N**umber = Parameter-Gruppennummer
PGN = PDU Format (PF) + PDU Source (PS)

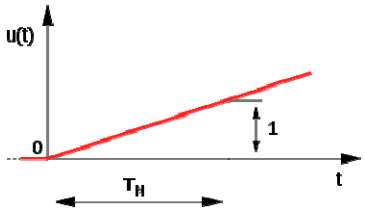
Die Parameter-Gruppennummer ist ein Begriff aus dem CAN-Protokoll SAE J1939. Sie fasst die Teiladressen PF und PS zusammen.

PID-Regler

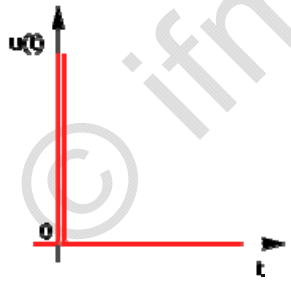
P = Proportional-Anteil
Der P-Regler besteht ausschließlich aus einem proportionalen Anteil der Verstärkung K_p. Mit seinem Ausgangssignal ist er proportional dem Eingangssignal.



I = Integral-Anteil
Ein I-Regler wirkt durch zeitliche Integration der Regelabweichung auf die Stellgröße mit der Gewichtung durch die Nachstellzeit T_N



D = Differential-Anteil
Der D-Regler reagiert nicht auf die Regelabweichung, sondern nur auf deren Änderungsgeschwindigkeit.



Piktogramm

Piktogramme sind bildhafte Symbole, die eine Information durch vereinfachte grafische Darstellung vermitteln.

→ Kapitel *Was bedeuten die Symbole und Formatierungen?*, → Seite 5

PL

Performance-**L**evel
Ist nach ISO 13849-1 eine Einstufung (PL a...e) der Fähigkeit von sicherheitsrelevanten Teilen einer Steuerung, eine Sicherheitsfunktion unter vorhersehbaren Bedingungen auszuführen.
→ Kapitel *Performance-Level PL*

PLr

Mit dem "erforderlichen Performance-Level" PL_r wird nach ISO 13849 die erforderliche Risikominderung für jede Sicherheitsfunktion erreicht.

Für jede gewählte Sicherheitsfunktion, die durch ein SRP/CS ausgeführt wird, muss ein PL_r festgelegt und dokumentiert werden. Die Bestimmung des PL_r ist das Ergebnis der Risikobeurteilung, bezogen auf den Anteil der Risikominderung durch die sicherheitsrelevanten Teile der Steuerung.

Pre-Op

Pre-Op = PRE-OPERATIONAL mode (engl.) = Zustand vor betriebsbereit

Betriebszustand eines CANopen-Teilnehmers. Nach dem Einschalten der Versorgungsspannung geht jeder Teilnehmer automatisch in diesem Zustand. Im CANopen-Netz können in diesem Modus nur SDOs und NMT-Kommandos übertragen werden, jedoch keine Prozessdaten.

prepared

prepared (engl.) = vorbereitet (auch: angehalten)

Betriebszustand eines CANopen-Teilnehmers. In diesem Modus werden nur NMT-Kommandos übertragen.

Programmiersprache, sicherheitsrelevant

Für sicherheitsrelevante Applikationen sollten nur folgende Programmiersprachen verwendet werden:

- Programmiersprache mit eingeschränktem Sprachumfang (LVL = **l**imited **v**ariability **l**anguage), kann vordefinierte, applikations-spezifische

© ifm electronic gmbh

Bibliotheksfunktionen kombinieren.
In CoDeSys sind das Kontaktplan KOP (Ladder Diagram LD) und Funktionsplan FUP (Function block diagram FBD).

- Programmiersprache mit nicht eingeschränktem Sprachumfang (FVL = **f**ull **v**ariability **l**anguage), kann einen großen Bereich von Funktionen kombinieren.
  Dazu gehören z.B. C, C++, Assembler. In CoDeSys ist das Strukturierter Text (ST).

► Strukturierter Text ist ausschließlich in gesonderten, zertifizierten Funktionen zu empfehlen, normalerweise in Embedded Software.

► Im "normalen" Applikations-Programm sollten nur KOP (LD) und FUP (FBD) eingesetzt werden. Damit sollen die folgenden Mindestanforderungen erfüllt werden können.

Generell werden folgende Mindestanforderungen an sicherheitsrelevante Applikations-Software (SRASW) gestellt:

► Programm modular und klar strukturieren. Folge: einfache Testbarkeit.

► Funktionen verständlich darstellen:
- für den Operator auf dem Bildschirm (Navigation),
- Lesbarkeit des späteren Dokumentationsausdrucks.

► Symbolische Variablen verwenden (keine IEC-Adressen).

► Variablennamen und Kommentare aussagekräftig formulieren.

► Einfache Funktionen verwenden (keine indirekte Adressierung, keine Variablenfelder).

► Defensiv programmieren.

► Leichtes Erweitern oder Anpassen des Programms ermöglichen.

Prozessabbild

Mit Prozessabbild bezeichnet man den Zustand der Ein- und Ausgänge, mit denen die SPS innerhalb eines Zyklusses arbeitet.

- Am Zyklus-Beginn liest die SPS die Zustände aller Eingänge in das Prozessabbild ein.
  Während des Zyklusses kann die SPS Änderungen an den Eingängen nicht erkennen.
- Im Laufe des Zyklusses werden die Ausgänge nur virtuell (im Prozessabbild) geändert.
- Am Zyklus-Ende schreibt die SPS die virtuellen Ausgangszustände auf die realen Ausgänge.

PWM

PWM = Puls-Weiten-Modulation

Via PWM kann ein (vom Gerät dazu befähigter) digitaler Ausgang mittels regelmäßiger, schneller Impulse eine beinahe analoge Spannung ausgeben. Bei dem PWM-Ausgangssignal handelt es sich um ein getaktetes Signal zwischen GND und Versorgungsspannung.

Innerhalb einer festen Periode (PWM-Frequenz) wird das Puls-/Pausenverhältnis variiert. Durch die angeschlossene Last stellt sich je nach Puls-/Pausenverhältnis der entsprechende Effektivstrom ein.
→ Kapitel *PWM-Signalverarbeitung*
→ Kapitel *Was macht ein PWM-Ausgang?*

R

Ratio

Ratio (lat.) = Verhältnis

Messungen können auch ratiometrisch erfolgen = Verhältnismessung. Das Eingangssignal erzeugt ein Ausgangssignal, das in einem bestimmten Verhältnis zu ihm liegt. Das bedeutet, ohne zusätzliche Referenzspannung können analoge Eingangssignale ausgewertet werden. Ein Schwanken der Versorgungsspannung hat auf diesen Messwert dann keinen Einfluss.
→ Kapitel *Zählerfunktionen*

RAW-CAN

RAW-CAN bezeichnet das reine CAN-Protokoll, das ohne ein zusätzliches Kommunikationsprotokoll auf dem CAN-Bus (auf ISO/OSI-Schicht 2) arbeitet. Das CAN-Protokoll ist international nach ISO 11898-1 definiert und garantiert zusätzlich in ISO 16845 die Austauschbarkeit von CAN-Chips.

© ifm electronic gmbh

redundant

Redundanz ist das Vorhandensein von mehr als den notwendigen Mitteln, damit eine Funktionseinheit eine geforderte Funktion ausführt oder damit Daten eine Information darstellen können.

Man unterscheidet verschiedene Arten der Redundanz:

- Die funktionelle Redundanz zielt darauf ab, sicherheitstechnische Systeme mehrfach parallel auszulegen, damit beim Ausfall einer Komponente die anderen den Dienst gewährleisten.
- Zusätzlich versucht man, die redundanten Systeme voneinander räumlich zu trennen. Dadurch minimiert man das Risiko, dass sie einer gemeinsamen Störung unterliegen.
- Schließlich verwendet man manchmal Bauteile unterschiedlicher Hersteller, um zu vermeiden, dass ein systematischer Fehler sämtliche redundanten Systeme ausfallen lässt (diversitäre Redundanz).

Die Software von redundanten Systemen sollte sich möglichst in den folgenden Aspekten unterscheiden:

- Spezifikation (verschiedene Teams),
- Spezifikationssprache,
- Programmierung (verschiedene Teams),
- Programmiersprache,
- Compiler.

remanent

Remanente Daten sind gegen Datenverlust bei Spannungsausfall geschützt.

Z.B. kopiert das Betriebssystem die remanenten Daten automatisch in einen Flash-Speicher, sobald die Spannungsversorgung unter einen kritischen Wert sinkt. Bei Wiederkehr der Spannungsversorgung lädt das Betriebssystem die remanenten Daten zurück in den Arbeitsspeicher.

Dagegen sind die Daten im Arbeitsspeicher einer Steuerung flüchtig und bei Unterbrechung der Spannungsversorgung normalerweise verloren.

Restrisiko

Das ist das verbleibende Risiko, nachdem Schutzmaßnahmen ergriffen wurden. Vor dem Restrisiko muss in Betriebsanleitungen und an der Maschine deutlich gewarnt werden.

Risiko

Als Risiko gilt die Kombination der Wahrscheinlichkeit des Eintritts eines Schadens und des Ausmaßes des Schadens.

Risikoanalyse

Kombination aus ...

- Festlegung der Grenzen der Maschine (Verwendungszweck, zeitliche Grenzen),
- Identifizierung der Gefährdung (Eingreifen von Personen, Betriebszustände der Maschine, vorhersehbarer Missbrauch) und
- der Risikoeinschätzung (Verletzungsgrad, Schadensumfang, Häufigkeit und Dauer der Gefahr, Eintrittswahrscheinlichkeit, Möglichkeit zur Vermeidung oder Begrenzung des Schadens).

Risikobeurteilung

Das ist die Gesamtheit des Verfahrens, das die Risikoanalyse und die Risikobewertung umfasst.

Nach Maschinenrichtlinie 2006/42/EG gilt: "Der Hersteller einer Maschine oder sein Bevollmächtigter hat dafür zu sorgen, dass eine Risikobeurteilung vorgenommen wird, um die für die Maschine geltenden Sicherheits- und Gesundheitsanforderungen zu ermitteln. Die Maschine muss dann unter Berücksichtigung der Ergebnisse der Risikobeurteilung konstruiert und gebaut werden." (→ Anhang 1, Allgemeine Grundsätze)

Risikobewertung

Das ist die auf der Risikoanalyse beruhende Beurteilung, ob die Ziele zur Risikominderung erreicht wurden.

© ifm electronic gmbh

ro

ro = read only (engl.) = nur lesen

Unidirektionale Datenübertragung: Daten können nur gelesen werden, jedoch nicht verändert.

RTC

RTC = **R**eal **T**ime **C**lock = Echtzeituhr

Liefert (batteriegepuffert) aktuell Datum und Uhrzeit. Häufiger Einsatz beim Speichern von Fehlermeldungsprotokollen.

Rückstellung, manuell

Die manuelle Rückstellung ist eine interne Funktion des SRP/CS zum anuellen Wiederherstellen einer oder mehrerer Sicherheitsfunktionen. Wird vor dem Neustart einer Maschine verwendet.

rw

rw = read/write (engl.) = lesen und schreiben

Bidirektionale Datenübertragung: Daten können sowohl gelesen als auch verändert werden.

S

SAE J1939

Das Netzwerkprotokoll SAE J1939 beschreibt die Kommunikation auf einem CAN-Bus in Nutzfahrzeugen zur Übermittlung von Diagnosedaten (z.B.Motordrehzahl, Temperatur) und Steuerungsinformationen.
→ CiA DS 402

Norm: "Recommended Practice for a Serial Control and Communications Vehicle Network"

Teil 2: "Agricultural and Forestry Off-Road Machinery Control and Communication Network"

Teil 3: "On Board Diagnostics Implementation Guide"

Teil 5: "Marine Stern Drive and Inboard Spark-Ignition Engine On-Board Diagnostics Implementation Guide"

Teil 11: "Physical Layer – 250 kBits/s, Shielded Twisted Pair"

Teil 13: "Off-Board Diagnostic Connector"

Teil 15: "Reduced Physical Layer, 250 kBits/s, Un-Shielded Twisted Pair (UTP)"

Teil 21: "Data Link Layer"

Teil 31: "Network Layer"

Teil 71: "Vehicle Application Layer"

Teil 73: "Application Layer – Diagnostics"

Teil 81: "Network Management Protocol"

Schaden

Als Schaden bezeichnet man eine physische Verletzung oder Schädigung der Gesundheit.

Schutzmaßnahme

Maßnahme zur vorgesehenen Minderung des Risikos, z.B.:
- fehlerausschließender Entwurf,
- technische Schutzmaßnahme (trennende Schutzeinrichtung),
- ergänzende Schutzmaßnahme (Benutzerinformation),
- persönliche Schutzausrüstung (Helm, Schutzbrille).

SCT

Bei CANopen-Safety überprüft die Sicherheits-Zykluszeit SCT (**S**afeguard **c**ycle **t**ime) die korrekte Funktion der periodischen Übertragung (Daten-Refresh) der SRDOs. Die Daten müssen innerhalb der eingestellten Zeit wiederholt worden sein, um gültig zu sein. Andernfalls signalisiert die empfangene Steuerung einen Fehler und geht in den sicheren Zustand (= Ausgänge abgeschaltet).

SD-Card

Eine SD Memory Card (Kurzform für **S**ecure **D**igital Memory Card; deutsch Sichere digitale Speicherkarte) ist ein digitales Speichermedium, das nach dem Prinzip der Flash-Speicherung arbeitet.

SDO

SDO = **S**ervice **D**ata **O**bject = Nachrichten-Objekt mit Servicedaten.

SDO ist eine Spezifikation für eine herstellerunabhängige Datenstruktur zum einheitlichen Datenzugriff. Dabei fordern

© ifm electronic gmbh

"Clients" die gewünschten Daten von "Servern" an. Die SDOs bestehen immer aus 8 Bytes. Längere Datenpakete werden auf mehrere Nachrichten verteilt.

Beispiele:

- Automatische Konfiguration aller Slaves über SDOs beim Systemstart.
- Auslesen der Fehlernachrichten aus dem Objektverzeichnis.

Jedes SDO wird auf Antwort überwacht und wiederholt, wenn sich innerhalb der Überwachungszeit der Slave nicht meldet.

Selbsttest

Testprogramm, das aktiv Komponenten oder Geräte testet. Das Programm wird durch den Anwender gestartet und dauert eine gewisse Zeit. Das Ergebnis davon ist ein Testprotokoll (Log-Datei), auf dem entnommen werden kann, was getestet wurde und ob das Ergebnis positiv oder negativ ist.

Sicherheitsfunktion

Der Ausfall einer Sicherheitsfunktion einer Maschine kann zum unmittelbar erhöhten Risiko führen. Der Konstrukteur einer solchen Maschine muss daher:
- einen Ausfall der Sicherheitsfunktion sicher verhindern,
- einen Ausfall der Sicherheitsfunktion rechtzeitig sicher erkennen,
- Maschine bei einem Ausfall der Sicherheitsfunktion rechtzeitig in einen sicheren Zustand bringen.

Sicherheits-Normentypen

Sicherheitsnormen auf dem Gebiet der Maschinen sind wie folgt strukturiert:

Typ-A-Normen (Sicherheits-Grundnormen) behandeln Grundbegriffe, Entwurfsleitsätze und allgemeine Aspekte, die auf Maschinen angewendet werden können. Beispiele: Terminologie Methodik (ISO 12100-1), Technische Prinzipien (ISO 12100-2), Risikobeurteilung (ISO 14121), ...

Typ-B-Normen (Sicherheits-Fachgrundnormen)
behandeln einen Sicherheitsaspekt oder eine Art von Schutzeinrichtungen, die für eine Reihe von Maschinen verwendet werden können.

- Typ-B1-Normen für bestimmte Sicherheitsaspekte. Beispiele: Sicherheits-Abstände (EN 294), Arm-/Hand-Geschwindigkeiten (EN 999), Sicherheitsbezogene Teile von Steuerungen (ISO 13849), Temperaturen, Lärm, ...
- Typ-B2-Normen für Schutzeinrichtrungen. Beispiele: Not-Aus-Schaltungen ((ISO 13850), Zweihand-Schaltungen, trennende oder berührungslos wirkende Schutzeinrichtungen (ISO 61496), ...

Typ-C-Normen (Maschinensicherheitsnormen) behandeln detaillierte Sicherheitsanforderungen an eine bestimmte Maschine oder Maschinengruppen.

SIL

Der Sicherheits-Integritätslevel SIL ist nach IEC 62061 eine Einstufung (SIL CL 1...4) der Sicherheitsintegrität der Sicherheitsfunktionen. Er dient der Beurteilung elektrischer/elektronischer/programmierbar elektronischer (E/E/PE)-Systeme in Bezug auf die Zuverlässigkeit von Sicherheitsfunktionen. Aus dem angestrebten Level ergeben sich die sicherheitsgerichteten Konstruktionsprinzipien, die eingehalten werden müssen, damit das Risiko einer Fehlfunktion minimiert werden kann.

Slave

Passiver Teilnehmer am Bus, antwortet nur auf Anfrage des →Masters. Slaves haben im Bus eine eindeutige und einmalige →Adresse.

SRDO

Über SRDOs (**S**afety-**R**elated **D**ata **O**bjects = *Sicherheitsrelevante Datenobjekte*) werden die sicheren Daten ausgetauscht. Ein SRDO besteht immer aus zwei CAN-Nachrichten mit unterschiedlichen Identifiern:

- Nachricht 1 enthält die Originalanwenderdaten,
- Nachricht 2 enthält die gleichen Daten, die aber bitweise invertiert werden.

© ifm electronic gmbh

SRP/CS

Safety-**R**elated **P**art of a **C**ontrol **S**ystem = Sicherheitsrelevanter Teil einer Steuerung

SRP/CS ist ein Teil einer Steuerung, das auf sicherheitsgerichtete Eingangssignale reagiert und sicherheitsgerichtete Ausgangssignale erzeugt. Die Kombination sicherheitsrelevanter Teile einer Steuerung beginnt an dem Punkt, an dem sicherheitsgerichtete Signale erzeugt werden (einschließlich Betätiger z.B. eines Positionsschalters) und endet an den Ausgängen der Leistungssteuerungselemente (einschließlich z.B. der Hauptkontakte eines Schützes).

SRVT

Die sicherheitsrelevante Objekt-Gültigkeitsdauer SRVT (**S**afety-**R**elated **O**bject **V**alidation **T**ime) sorgt bei CANopen-Safety dafür, dass die Zeit zwischen den SRDO-Nachrichten-Paaren eingehalten wird:

Nur wenn die redundante, invertierte Nachricht innerhalb der eingestellten Zeit SRVT nach der Original-Nachricht übertragen wurde, sind die damit übertragenen Daten gültig. Andernfalls signalisiert die empfangende Steuerung einen Fehler und geht in den sicheren Zustand (= Ausgänge abgeschaltet).

Steuerungskonfiguration

Bestandteil der CoDeSys-Bedienoberfläche.

► Programmierer teilt dem Programmiersystem mit, welche Hardware programmiert werden soll.

> CoDeSys lädt die zugehörigen Bibliotheken.

> Lesen und schreiben der Peripherie-Zustände (Ein-/Ausgänge) ist möglich.

Symbole

Piktogramme sind bildhafte Symbole, die eine Information durch vereinfachte grafische Darstellung vermitteln.

→ Kapitel *Was bedeuten die Symbole und Formatierungen?*, → Seite 5

Systemvariable

Variable, auf die via IEC-Adresse oder Symbolname aus der SPS zugegriffen werden kann.

T

Target

Das Target gibt das Zielsystem an, auf dem das SPS-Programm laufen soll. Im Target sind die Dateien (Treiber und ggf. spezifische Hilfedateien) enthalten, die zum Programmieren und Parametrieren erforderlich sind.

TCP

Das **T**ransmission **C**ontrol **P**rotocol ist Teil der Protokollfamilie TCP/IP. Jede TCP/IP-Datenverbindung hat einen Sender und einen Empfänger. Dieses Prinzip ist eine verbindungsorientierte Datenübertragung. In der TCP/IP-Protokollfamilie übernimmt TCP als verbindungsorientiertes Protokoll die Aufgabe der Datensicherheit, der Datenflusssteuerung und ergreift Maßnahmen bei einem Datenverlust. (vgl.: →UDP)

Template

Template (englisch = Schablone)

Ist eine Vorlage, die mit Inhalten gefüllt werden kann.
Hier: Eine Struktur von vorkonfigurierten Software-Elementen als Basis für ein Applikations-Programm.

Testrate rt

Die Testrate r_t ist die Häufigkeit der automatischen Tests, um Fehler in einem SRP/CS rechtzeitig zu bemerken.

U

Überwachung

Die Überwachung ist eine Sicherheitsfunktion, die sicherstellt, dass eine Schutzmaßnahme eingeleitet wird, sobald Folgendes eintritt:

- Die Fähigkeit eines Bauteils oder eines Elements, seine Funktion auszuführen, wird vermindert.

© ifm electronic gmbh

- Die Betriebsbedingungen werden so verändert, dass das resultierende Risiko steigt.

UDP

UDP (**U**ser **D**atagram **P**rotocol) ist ein minimales, verbindungsloses Netzprotokoll, das zur Transportschicht der Internetprotokollfamilie gehört. Aufgabe von UDP ist es, Daten, die über das Internet übertragen werden, der richtigen Applikation zukommen zu lassen.

Derzeit sind Netzwerkvariablen auf Basis von CAN und UDP implementiert. Die Variablenwerte werden dabei auf der Basis von Broadcast-Nachrichten automatisch ausgetauscht. In UDP sind diese als Broadcast-Telegramme realisiert, in CAN als PDOs. Diese Dienste sind vom Protokoll her nicht bestätigte Dienste, d.h. es gibt keine Kontrolle, ob die Nachricht auch beim Empfänger ankommt. Netzwerkvariablen-Austausch entspricht einer "1-zu-n-Verbindung" (1 Sender zu n Empfängern).

V

Verwendung, bestimmungsgemäß

Das ist die Verwendung eines Produkts in Übereinstimmung mit den in der Anleitung bereitgestellten Informationen.

W

Watchdog

Der Begriff Watchdog (englisch; Wachhund) wird verallgemeinert für eine Komponente eines Systems verwendet, die die Funktion anderer Komponenten beobachtet. Wird dabei eine mögliche Fehlfunktionen erkannt, so wird dies entweder signalisiert oder geeignete Programm-Verzweigungen eingeleitet. Das Signal oder die Verzweigungen dienen als Auslöser für andere kooperierende Systemkomponenten, die das Problem lösen sollen.

wo

wo = write only (engl.) = nur schreiben

Unidirektionale Datenübertragung: Daten können nur verändert werden, jedoch nicht gelesen.

Z

Zustand, sicher

Der Zustand einer Maschine gilt als sicher, wenn von ihr keine Gefährdung mehr ausgeht. Dies ist meist der Fall, wenn alle gefahrbringenden Bewegungsmöglichkeiten abgeschaltet sind und nicht unerwartet wieder anlaufen können.

Zykluszeit

Das ist die Zeit für einen Zyklus. Das SPS-Programm läuft einmal komplett durch.

Je nach ereignisgesteuerten Verzweigungen im Programm kann dies unterschiedlich lange dauern.

ifm electronic gmbh

14 Index

A

Abbildungen ... 310
Abfrage des Slave-Gerätetyps ... 115
Abgrenzung zu anderen CANopen-Bibliotheken ... 105
Adresse ... 333
Allgemeine Dateifunktionen ... 196
Allgemeine Informationen ... 135
Allgemeine Übersicht ... 323
Allgemeines zu CAN ... 83
Allgemeines zu CANopen mit CoDeSys ... 103
Analoge Werte anpassen ... 180
Ändern der PDO-Eigenschaften zur Laufzeit ... 134
Anforderungsrate rd ... 333
Angaben zum Gerät ... 9
Angaben zur Software (Linux | CoDeSys v2.3) ... 9
Anhang ... 289
Anleitung ... 333
Applikations-Software ... 333
Arbeitsreihenfolge ... 78
Architektur ... 333
ARRAY_TO_DWORD ... 223
ARRAY_TO_WORD ... 222
Asynchrone Dateifunktionen ... 233
Aufbau des COB-ID ... 292
Aufbau einer EMCY-Nachricht ... 173
Aufbau einer Fehlernachricht ... 173
Aufbau von CANopen-Meldungen ... 291
Ausfall ... 333
Ausfall, gefahrbringend ... 333
Ausfall, systematischer ... 333
Automatische Konfiguration von Slaves ... 116

B

Baud ... 333
Begrenzungen beim PDM360compact ... 71
Begrenzungen und Programmierhinweise ... 69
Beispiel
- CANx_MASTER_SEND_EMERGENCY ... 146
- CANx_MASTER_STATUS ... 150
- CANx_SLAVE_SEND_EMERGENCY ... 159
- INPUT_INT mit 3 Eingabeelementen ... 275
- Variablenliste ... 132
- Verkleinern eines Pixelbildes ... 320

Beispiel für ein Objektverzeichnis ... 128
Beispiele
- PDM_MAIN_MAPPER ... 257

Beschreibung der CAN-Standardbausteine ... 96
Besonderheiten bei Netzwerkvariablen ... 140
Bestimmungsgemäße Verwendung ... 333
Betriebsart der LED-Kette ... 268
Betriebsdauer, mittlere ... 334
Betriebssystem ... 334
Bibliothek Instrumente ... 264
Bibliotheken ... 77
Bibliotheken für CANopen ... 141
Bild übertragen per FTP ... 17
Bild umrechnen / skalieren ... 73
Bildgröße Vektorgrafik / Pixelgrafik ... 320
Bitmap-Grafiken anpassen ... 321
Bootloader ... 334
Bootup-Nachricht ... 296
Bus ... 334
Busleitungslänge ... 90
BYTE_ARRAY_TO_CSV ... 226
BYTE_ARRAY_TO_STRING ... 227

C

CAN ... 334
CAN Download-ID einstellen ... 26
CAN einsetzen ... 83
CAN Parameter
- Alle SDOs erzeugen ... 112
- Automatisch starten ... 110
- Baudrate ... 108
- Communication Cycle ... 113
- Communication Cycle Period / Sync. Window Length ... 109
- DCF schreiben ... 112
- Emergency Telegram ... 113
- Heartbeat ... 110
- Knoten zurücksetzen ... 112
- Nicht initialisieren ... 112
- Nodeguarding- / Heartbeat-Einstellungen ... 113
- Node-ID ... 110, 112
- Optionales Gerät ... 112
- Sync. COB-ID ... 109

CAN_1_RECEIVE ... 97
CAN_1_TRANSMIT ... 99
CAN_x_RECEIVE ... 97
CAN_x_TRANSMIT ... 99
CAN1_MASTER_EMCY_HANDLER ... 142
CAN1_MASTER_SEND_EMERGENCY ... 144
CAN1_MASTER_STATUS ... 147
CAN1_SDO_READ ... 164
CAN1_SDO_WRITE ... 166
CAN1_SLAVE_BAUDRATE ... 153
CAN1_SLAVE_EMCY_HANDLER ... 155
CAN1_SLAVE_NODEID ... 154
CAN1_SLAVE_SEND_EMERGENCY ... 157
CAN1_SLAVE_STATUS ... 160
CAN-Buspegel ... 89
CAN-Datenaustausch ... 92
CAN-Fehler ... 171

© ifm electronic gmbh

CAN-Fehler und Fehlerbehandlung 170
CAN-ID 93
CANopen Begriffe und Implementation 104
CANopen Error-Code 301
CANopen Netzwerk-Konfiguration, Status- und Fehlerbehandlung 102
CANopen-Master 105
Register [CAN-Parameter] 108
CANopen-Netzwerkvariablen 135
CANopen-Netzwerkvariablen konfigurieren 136
CANopen-Slave 126
Register [CAN Parameter] 112
CANopen-Slave konfigurieren 127
CANopen-Slaves einfügen und konfigurieren 111
CANopen-Status des Knotens 121, 300
CANopen-Tabellen 290
CANopen-Unterstützung durch CoDeSys 103
CAN-Schnittstelle prüfen 22
CAN-Schnittstellen 85
CAN-Stack 334
CANx_MASTER_EMCY_HANDLER 142
CANx_MASTER_SEND_EMERGENCY 144
CANx_MASTER_STATUS 147
CANx_SDO_READ 164
CANx_SDO_WRITE 166
CANx_SLAVE_BAUDRATE 153
CANx_SLAVE_EMCY_HANDLER 155
CANx_SLAVE_NODEID 154
CANx_SLAVE_SEND_EMERGENCY 157
CANx_SLAVE_STATUS 160
CCF 334
CiA 334
CiA DS 304 334
CiA DS 401 334
CiA DS 402 334
CiA DS 403 334
CiA DS 404 334
CiA DS 405 334
CiA DS 406 335
CiA DS 407 335
COB-ID 335
CoDeSys 335
CoDeSys-CANopen-Bibliotheken 327
CoDeSys-Kommunikationsparameter einstellen 24
CoDeSys-Kommunikationsparameter für die Ethernet-Schnittstelle einstellen 34
CoDeSys-Kommunikationsparameter für die serielle Schnittstelle einstellen 35
CoDeSys-Visualisierungs-Elemente 74
CONTROL_ANALOGCLOCK 266
CPU-Frequenzen 69
CRC 335
CURRENTVISU_TO_NO 261

D

Das Objektverzeichnis des CANopen-Masters 124
Das Penalty-Konzept 247
Dateien für Betriebssystem / Laufzeitsystem 325
Dateien und Bibliotheken im Gerät installieren 322
Dateien verwalten 194
Dateifunktionen 194
Dateifunktionen für CSV-Dateien 205
Dateifunktionen für die Alarmverwaltung 215
Dateifunktionen für Parameterdateien 209
Dateifunktionen für Textdateien 201
Daten empfangen 95
Daten senden 95
Daten verwalten 185
Datenzugriff und Datenprüfung 190
DC 335
DELETE_FILE 197
Demo-Programme für Controller 53
Demo-Programme für PDM und BasicDisplay 55
Der Master zur Laufzeit 115
DEVICE_KERNEL_VERSION2 191
DEVICE_RUNTIME_VERSION 192
DEVICE_TS 193
Diagnose 335
Diagnose-Deckungsgrad 336
Dither 336
diversitär 336
DRAM 336
DTC 336
DUMP_FILESYSTEM 231
DWORD_TO_ARRAY 225

E

Echtzeituhr (RTC) einstellen 20
ECU 336
EDS-Datei 337
Ein CANopen-Projekt erstellen 107
Ein-/Ausgangs-Funktionen 178
Eingabefelder in Visualisierung bearbeiten 271
Eingangswerte verarbeiten 178
Einstellen der Knotennummer und der Baud-Rate eines CANopen-Slaves 134
Einstellungen in den globalen Variablenlisten 137
Einstellungen in den Zielsystemeinstellungen 136
Embedded Software 337
EMCY 337
EMCY-Fehlercode 174
Emergency-Messages durch das Applikations-Programm senden 134
Emergency-Nachrichten 301
Empfangen von Emergency-Messages 116
Empfehlungen für Bedienoberflächen 305
Empfehlungen für Task-Konfigurationen 242

© ifm electronic gmbh

Empfehlungen zur nutzerfreundlichen Produktgestaltung 305
EMV 337
Erste Schritte 57
Erstfehler-Eintrittszeit 337
Ethernet 337
Ethernet testen über Hyper Terminal 32
Ethernet testen über PING-Funktion 31
Ethernet-Schnittstelle
IP-Adresse einstellen 13
Subnetzmaske einstellen 15
Ethernet-Verbindung testen 31
Ethernet-Verbindung und Spannungsversorgung herstellen 66
EUC 337

F

Farben 309
FB, FUN, PRG in CoDeSys 76
Fehlanwendung 337
Fehlende Bibliotheken einfügen 57
Fehler 338
Fehler und Störungen beheben 289
Fehlertelegramm 171
Fehler-Toleranzzeit 338
Fehlerzähler 171
FiFo 338
FILE_AVAIL 235
FILE_MAX 236
FILE_READ 237
FILE_WRITE 239
Firmware 338
Flash-Speicher 338
FMEA 338
FRAM 339
Funktion von Tasten, Drehgeber und LEDs prüfen 21
Funktionale Sicherheit 339
Funktionalität der CANopen-Slave-Bibliothek 126
Funktions-Code / Predefined Connectionset 293

G

Gebrauchsdauer Tm 339
Gebrauchstauglichkeit prüfen 307
Gefährdung 339
Gerätefehler signalisieren 174
Geräteparameter einstellen (Setup) 11
Geräte-Update auf neue Software-Version 63
Geräte-Update durchführen 66
Geräte-Update vorbereiten 63
GET_PENALTY 242
Globale Variable dieses Programms 253
Globale Variablenliste
Bestätigter Transfer 139
Ereignisgesteuerte Übertragung 139
Lesen 139
Netzwerktyp 138
Prüfsumme übertragen 139
Schreiben 139
Übertragung bei Änderung 139
Variablen packen 138
Variablenlistenkennung (COB-ID) 138
Zyklische Übertragung 139
Grundeinstellungen
EDS-Datei generieren 127
Name der Updatetask 127
Name des Busses 127
Grundlegende Informationen zu Bitmap-Grafiken 319
Grundsätzliches 304

H

Heartbeat 339
Heartbeat vom Master an die Slaves 116
Herstellerspezifische Informationen 177
Hilfsfunktionen für Arrays 221
Hinweise 93
HMI 339
Hochlauf der CANopen-Slaves 120
Hochlauf des CANopen-Masters 118
Hochlauf des Netzwerks ohne [Automatisch starten] 121

I

ID - Identifier 339
Identifier 173
IDs (Adressen) in CANopen 104, 290
IEC-User-Zyklus 339
ifm weltweit • ifm worldwide • ifm à l'échelle internationale 357
ifm-Bibliothek für den CANopen-Master 141
ifm-Bibliothek für den CANopen-Slave 152
ifm-CANopen-Bibliotheken 101
ifm-CANopen-Hilfsbibliotheken Master/Slave 326
ifm-Demo-Programme 53
ifm-Gerätebibliotheken 326
ifm-Maintenance-Tool nutzen 79
INIT_VAR_RETAIN_DINT 285
INIT_VAR_RETAIN_INT 283
INIT_VAR_RETAIN_REAL 287
Initialisieren des Netzwerks mit RESET_ALL_NODES 122
INPUT_DINT 277
INPUT_INT 272
INPUT_REAL 280
Interne Parameter 274, 279, 282
Internet-Protokoll auf dem Computer einstellen 30
IP-Adresse 339
IP-Adresse des Server-PC einstellen 64
ISO 10646 _ Informationstechnik – Universeller Mehrfach-8-bit-codierter Zeichensatz (UCS) 316
ISO 11898 340

© ifm electronic gmbh

ISO 11992 340
ISO 13406 _ Ergonomische Anforderungen für Tätigkeiten an optischen Anzeigeeinheiten in Flachbauweise 317
ISO 13407 _ Benutzer-orientierte Gestaltung interaktiver Systeme 317
ISO 16845 340
ISO 20282 _ Bedienungsfreundlichkeit von Produkten des täglichen Gebrauchs 318
ISO 7001 _ Graphische Symbole zur Information der Öffentlichkeit 311
ISO 9126 _ Qualitätsmerkmale für Software-Produkte 312
ISO 9241 _ Ergonomie der Mensch-System-Interaktion 314
ISO 9241-11 _ Anforderungen an die Gebrauchstauglichkeit 315
ISO 9241-110 _ Grundsätze der Dialoggestaltung 315

K

Kategorie (CAT) 340
Kennen Sie die künftigen Nutzer? 306
Klemme 15 340
Konfiguration aller fehlerfrei detektierten Geräte 115
Konfigurationen 11
Kulturelle Details sind oft nicht übertragbar 309

L

Lebensdauer, mittlere 340
LED 340
LED ansteuern 249
LED, Buzzer, Visualisierung 249
Leistungsgrenzen des Geräts 69
Leitungsquerschnitte 91
Leserichtung 310
Link 340
LOG_TIMESTAMP 232
LSB 340

M

MAC-ID 340
Man unterscheidet folgende Fehler: 173
Master 341
MMI 341
MRAM 341
MSB 341
MTBF 341
MTTF 341
MTTFd 341
Muting 341

N

Netzaufbau 88
Netzwerk starten 117
Netzwerk-Management (NMT) 297
Netzwerk-Management-Kommandos 297
Netzwerkvariablen 82
Netzwerkzustände 118
NMT 341
NMT-Status 297
NMT-Status für CANopen-Master 119, 298
NMT-Status für CANopen-Slave 120, 299
Node 341
Node Guarding 341
Nodeguarding mit Lifetime-Überwachung 116
NORM_DINT 181
NORM_REAL 183

O

Obj / Objekt 342
Objekt 0x1001 (Error-Register) 176, 303
Objekt 0x1003 (Error Field) 174
Objektverzeichnis 342
OBV 342
OPEN_PCMCIA 198
OPEN_PCMCIA_VFAT 199
operational 342
OPT_SCHEDULER 244
Ordner-Struktur, allgemein 46

P

PAGENO_TO_CURRENTVISU 263
Parameter der internen Strukturen 149
PC-Karte 342
PCMCIA-Karte 342
PDM 342
PDM_FILE_VERSION 230
PDM_LED_OFF 250
PDM_MAIN 252
PDM_MAIN einsetzen 254
PDM_MAIN_MAPPER 255
PDM_PAGECONTROL 259
PDM_READ_RTC 186
PDM_SET_RTC 188
PDM-Setup
  Startbild des PDM ändern 19
PDM-Setup verlassen, Gerät neu starten 27
PDO 342
PDO-Mapping
  Eigenschaften 114
  Einfügen 113
PDU 342
Performance-Level 342
PES 342
PGN 343
Physikalische Anbindung des CAN 88
PID-Regler 343
Piktogramm 343
PL 343
PLC-Programm erstellen 61

© ifm electronic gmbh

PLr 343
Pre-Op 343
prepared 343
Programme und Funktionen in den Ordnern der Templates 47
Programmierhinweise für CoDeSys-Projekte 75
Programmierschnittstellen 28
Programmiersprache, sicherheitsrelevant 343
Programmiersystem einrichten 39
Programmiersystem manuell einrichten 39
Programmiersystem über Templates einrichten 44
Programmierung über die Ethernet-Schnittstelle 29
Programmierung über die serielle Schnittstelle RS232 35
Projekt im Gerät speichern 62
Projekt mit weiteren Funktionen ergänzen 50
Prozessabbild 344
PWM 344

R

Ratio 344
RAW-CAN 344
READ_ALARM 216
READ_COM_NODEID 168
READ_CSV_8BYTE 206
READ_PARAMETER_WORD 210
READ_SINGLE_PARAMETER 213
READ_TEXTFILE 202
READ_TEXTLINE 203
READBUFFER_TO_STRING 228
redundant 345
Register [CAN-Einstellungen] 130
Register [Default PDO-Mapping] 131
Register [Grundeinstellungen] 127
Register [PDO-Mapping empfangen] und [PDO-Mapping senden] 113
Register [Service Data Objects] 114
remanent 345
Reset aller konfigurierten Slaves am Bus beim Systemstart 115
Restrisiko 345
Retain-Variable 81
Richtlinien und Normen 311
Risiko 345
Risikoanalyse 345
Risikobeurteilung 345
Risikobewertung 345
ro 346
RTC 346
Rücklesen von Retain-Variablen 82
Rückstellung, manuell 346
rw 346

S

SAE J1939 346
SCALE_LED_GRAF 267
SCALE_METER 269
Schaden 346
Schutzmaßnahme 346
SCT 346
SD-Card 346
SDO 346
SDO-Abbruch-Code 295
SDO-Kommando-Bytes 294
Selbsttest 347
Serielle Schnittstelle einstellen 25
SET_COM_NODED 169
Setup starten 12
Sicherheitsfunktion 347
Sicherheitshinweise 7
Sicherheits-Normentypen 347
Sichern von Retain-Variablen 81
SIL 347
Slave 347
Slave-Informationen 150
Software für CAN und CANopen 87
Speichern 78
spezielle ifm-Bibliotheken 328
Sprache als Hindernis 307
SPS-Zyklus optimieren 241
SRDO 347
SRP/CS 348
SRVT 348
Start aller fehlerfrei konfigurierten Slaves 116
Startbild des PDM ändern 17
Starten des Netzwerks mit GLOBAL_START 122
Starten des Netzwerks mit START_ALL_NODES 122
Steuerungskonfiguration 10, 348
Steuerungskonfiguration aktivieren (z.B. CR1051) 41
Steuerungskonfigurations-Datei 325
STORE_ALARM 218
Struktur der Alarmmeldungen 216, 219
Struktur der Visualisierungen in den Templates 49
Struktur Emergency_Message 151
Struktur Knoten-Status 150
Symbol-Dateien 38
Symbole 310, 348
SYNC 200
Systembeschreibung 9
System-Konfiguration 87
Systemvariable 348
Systemzeit lesen / schreiben 185

T

Target 348
Target einrichten 39
Target-Datei 325
TCP 348

© ifm electronic gmbh

Technisches zu CANopen 101
Teilnehmer bus-off 172
Teilnehmer fehleraktiv 172
Teilnehmer fehlerpassiv 172
Template 348
Test
CAN-Telegramme empfangen (RX) 23
CAN-Telegramme senden (TX) 22
Testrate rt 348
Texte 75
TFTP-Programm PumpKIN installieren und konfigurieren 64
TOGGLE 179
Topologie 84

U

Über die ifm-Templates 46
Über diese Anleitung 5
Übersicht CANopen EMCY-Codes (CR105n, CR106n) 177
Übersicht CANopen Error-Codes 175, 302
Übersicht der verwendeten Dateien und Bibliotheken 322
Überwachung 348
UDP 349
Update-Dateien vorbereiten 63
Update-Software in das Gerät laden 67

V

Variable 80
Verändern des Standard-Mappings durch Master-Konfiguration 133
Verfügbare CAN-Schnittstellen und CAN-Protokolle 85
Verhalten des Watchdog 70
Verwendung, bestimmungsgemäß 349
Visualisierung erstellen 59
Visualisierung verwalten 251
Visualisierungen im Gerät 304
Visualisierungsgrenzen 72

W

Was bedeuten die Symbole und Formatierungen? 5
Was ist ein Scheduler? 245
Was macht der 246
Was macht der 3S-Task-Scheduler? 245
Was wird benötigt? 63
Watchdog 349
Weitere Hilfsfunktionen 229
Weitere ifm-Bibliotheken zu CANopen 163
Welche Vorkenntnisse sind notwendig? 8
Wichtig! 7
Wie ist diese Anleitung aufgebaut? 6
wo 349
WORD_TO_ARRAY 224
Wozu dienen die einzelnen Dateien und Bibliotheken? 325
WRITE_CSV_8BYTE 207
WRITE_PARAMETER_WORD 212
WRITE_SINGLE_PARAMETER 214

Z

Zeichensätze (Fonts) installieren 36
Zugriff auf den CANopen-Slave zur Laufzeit 134
Zugriff auf den Status des CANopen-Masters 123
Zugriff auf die OD-Einträge vom Applikations-Programm 134
Zugriff auf die Strukturen zur Laufzeit der Applikation 151
Zusammenfassung CAN / CANopen 94
Zustand, sicher 349
Zyklisches Senden der SYNC-Message 116
Zykluszeit 349
Zykluszeit steuern 241

© ifm electronic gmbh

15 ifm weltweit • ifm worldwide • ifm à l'échelle internationale

Stand: 2010-10-08 ifm electronic – close to you!

http://www.ifm.com • E-Mail: info@ifm.com

Service-Hotline: 0800 16 16 16 4 (nur Deutschland, Mo...Fr, 07.00...18.00 Uhr)

ifm Niederlassungen • Sales offices • Agences

D	ifm electronic gmbh Vertrieb Deutschland Niederlassung Nord • 31135 Hildesheim • Tel. 0 51 21 / 76 67-0 Niederlassung West • 45128 Essen • Tel. 02 01 / 3 64 75 -0 Niederlassung Mitte-West • 58511 Lüdenscheid • Tel. 0 23 51 / 43 01-0 Niederlassung Süd-West • 64646 Heppenheim • Tel. 0 62 52 / 79 05-0 Niederlassung Baden-Württemberg • 73230 Kirchheim • Tel. 0 70 21 / 80 86-0 Niederlassung Bayern • 82178 Puchheim • Tel. 0 89 / 8 00 91-0 Niederlassung Ost • 07639 Tautenhain • Tel. 0 36 601 / 771-0 ifm electronic gmbh • Friedrichstraße 1 • 45128 Essen
A	ifm electronic gmbh • 1120 Wien • Tel. +43 16 17 45 00
AUS	ifm efector pty ltd. • Mulgrave Vic 3170 • Tel. +61 3 00 365 088
B, L	ifm electronic N.V. • 1731 Zellik • Tel. +32 2 / 4 81 02 20
BR	ifm electronic Ltda. • 03337-000, Sao Paulo SP • Tel. +55 11 / 2672-1730
CH	ifm electronic ag • 4 624 Härkingen • Tel. +41 62 / 388 80 30
CN	ifm electronic Co. Ltd. • 201210 Shanghai • Tel. +86 21 / 5027 8559
CND	ifm efector Canada inc. • Oakville, Ontario L6K 3V3 • Tel. +1 800-441-8246
CZ	ifm electronic spol. s.r.o. • 25243 Průhonice • Tel. +420 267 990 211
DK	ifm electronic a/s • 2605 BROENDBY • Tel. +45 70 20 11 08
E	ifm electronic s.a. • 08820 El Prat de Llobregat • Tel. +34 93 479 30 80
F	ifm electronic s.a. • 93192 Noisy-le-Grand Cedex • Tél. +33 0820 22 30 01
FIN	ifm electronic oy • 00440 Helsinki • Tel . +358 75 329 5000
GB, IRL	ifm electronic Ltd. • Hampton, Middlesex TW12 2HD • Tel. +44 208 / 213-0000
GR	ifm electronic Monoprosopi E.P.E. • 15125 Amaroussio • Tel. +30 210 / 6180090
H	ifm electronic kft. • 9028 Györ • Tel. +36 96 / 518-397
I	ifm electronic s.a. • 20041 Agrate-Brianza (MI) • Tel. +39 039 / 68.99.982
IL	Astragal Ltd. • Azur 58001 • Tel. +972 3 -559 1660
IND	ifm electronic India Branch Office • Kolhapur, 416234 • Tel. +91 231-267 27 70
J	efector co., ltd. • Togane-shi, Chiba 283-0826 • Tel. +81 475-50-3003
MAL	ifm electronic Pte. Ltd • 80250 Johor Bahru Johor • Tel. +60 7 / 331 5022
MEX	ifm efector S. de R. L. de C. V. • Monterrey, N. L. 64630 • Tel. +52 81 8040-3535
N	Sivilingeniør J. F. Knudtzen A/S • 1396 Billingstad • Tel. +47 66 / 98 33 50
NL	ifm electronic b.v. • 3843 GA Harderwijk • Tel. +31 341 / 438 438
P	ifm electronic s.a. • 4430-208 Vila Nova de Gaia • Tel. +351 223 / 71 71 08
PL	ifm electronic Sp. z o.o. • 40-524 Katowice • Tel. +48 32-608 74 54
RA, ROU	ifm electronic s.r.l. • 1107 Buenos Aires • Tel. +54 11 / 5353 3436
ROK	ifm electronic Ltd. • 140-884 Seoul • Tel. +82 2 / 790 5610
RP	Gram Industrial, Inc. • 1770 Mantilupa City • Tel. +63 2 / 850 22 18
RUS	ifm electronic • 105318 Moscow • Tel. +7 495 921-44-14
S	ifm electronic a b • 512 60 Överlida • Tel. +46 325 / 661 500
SGP	ifm electronic Pte. Ltd. • Singapore 609 916 • Tel. +65 6562 8661/2/3
SK	ifm electronic s.r.o. • 835 54 Bratislava • Tel. +421 2 / 44 87 23 29
THA	Sang Chai Meter Co., Ltd. • Bangkok 10 400 • Tel. +66 2 / 616 80 51
TR	ifm electronic Ltd. Sti. • 34381 Sisli/Istanbul • Tel. +90 212 / 210 50 80
UA	TOV ifm electronic • 02660 Kiev • Tel. +380 44 501 8543
USA	ifm efector inc. • Exton, PA 19341 • Tel. +1 610 / 5 24-2000
ZA	ifm electronic (Pty) Ltd. • 0157 Pretoria • Tel. +27 12 345 44 49

Technische Änderungen behalten wir uns ohne vorherige Ankündigung vor.
We reserve the right to make technical alterations without prior notice.
Nous nous réservons le droit de modifier les données techniques sans préavis.

© ifm electronic gmbh